Set-Stat-Map: Visualizing Spatial Data with Mixed Numeric and Categorical Attributes

A Thesis Submitted to the
College of Graduate and Postdoctoral Studies
in Partial Fulfillment of the Requirements
for the degree of Master of Science
in the Department of Computer Science
University of Saskatchewan
Saskatoon

By
Shisong Wang

Permission to Use

In presenting this thesis in partial fulfilment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Computer Science
176 Thorvaldson Building
110 Science Place
University of Saskatchewan
Saskatoon, Saskatchewan
Canada
S7N 5C9

Or

Dean
College of Graduate and Postdoctoral Studies
University of Saskatchewan
116 Thorvaldson Building, 110 Science Place
Saskatoon, Saskatchewan S7N 5C9
Canada

Abstract

Multi-attribute datasets are common and appear in many important scenarios for data analytics. Such data can be complex and thus difficult to understand directly without using visualization techniques. Existing visualizations for multi-attribute datasets are often designed based on attribute types, i.e., whether the attributes are categorical or numerical. Parallel Coordinates and Parallel Sets are two well-known techniques to visualize numerical and categorical data, respectively. However, visualization for mixed data types appears to be challenging. A common strategy to visualize mixed data is to use multiple information-linked views, e.g., Parallel Coordinates are often augmented with maps to explore spatial data with numeric attributes. In this paper, we design visualizations for mixed data types, where the dataset may include numerical, categorical, and spatial attributes.

The proposed solution Set-Stat-Map is a harmonious combination of three interactive components: Parallel Sets (visualizes sets determined by the combination of categories or numeric ranges), statistics columns (visualizes numerical summaries of the sets), and a dataset-specified map view (geospatial map view for spatial information, heatmap for pairwise information, etc.). We also augment the Parallel Sets view in two main ways: First, we impose textures on top of colors, which are spread into the other views, to enhance users' capability of analyzing distributions of pairs of attribute combinations. Second, we limit the number of sets for each axis to a small number by merging some of them into one and limit the sizes of the merged sets to improve the rendering performance as well as to reduce users' cognitive loads.

We demonstrate the use of Set-Stat-Map using different types of datasets: a meteorological dataset (CFSR), an online vacation rental dataset (Airbnb), and a software developer community dataset (Stack-Overflow). We provide design guidelines based on the results of the analysis of the performance from both visual analytics and scalability aspects. To examine the usability of the system, we collaborated with meteorologists, which reveals both challenges and opportunities for Set-Stat-Map to be used for real-life visual analytics.

Acknowledgements

This project would not have been possible without the support of many people. I would like to express my heartfelt gratitude to my supervisors Dr. Debajoyti Mondal, Dr. Chanchal K. Roy, and Dr. Kevin Schneider for their constant guidance, advice, motivation, and extraordinary patience. They have been extremely supportive and helpful to me both academically and personally.

I am thankful to Dr. Sara Sadri for her valuable feedback on the system iterations during the development.

Special thanks to all members of the Visualization, Geometry, and Algorithms Lab and all the staff members of the Department of Computer Science who helped me along the way.

I am grateful to the Global Water Futures and the Department of Computer Science for their generous financial support through scholarships that helped me to focus on my thesis work.

Finally, I would like to acknowledge the support of my family and friends for helping me along the journey of studying in Canada.

This thesis is dedicated to my mother, father, grandmother, grandfather, and aunt for their unconditional love and support.

Contents

List of Tables

List of Figures

1 Introduction

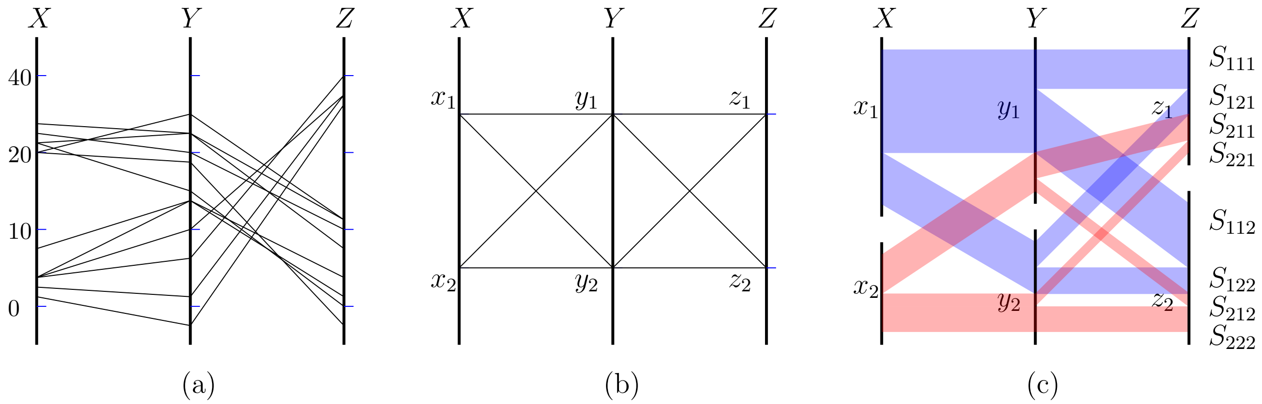
The availability of big data requires people to deal with a large amount of information to make decisions in business and industries. Data visualization is a technique to help people understand specific data through visual observation, which reduces noise and highlights the information that people actually want. Therefore, by using visualization techniques, people can understand data easily and efficiently.

Multi-attribute datasets are common in real-life data analytics scenarios such as datasets of student performances and demographics, hotel attributes and reviews, medical examination reports, etc. Such data are often explored visually due to humans' ability to rapidly comprehend and detect patterns in visual scenes. However, the preferred choice of visualization varies depending on the data attributes. A traditional approach to visually explore numeric data is to use statistical charts or a scatter plot matrix, where each entry contains a scatterplot for the corresponding row and column attributes. Here we assume a numeric attribute to be either continuous or to have many discrete values over a range. For categorical data, where an attribute may have only a few discrete values or categories, a traditional way is to visualize a frequency matrix, where each entry contains a value representing the number of data records in the row and column category. More recent approaches to examine multi-attribute datasets are Parallel Coordinates [39] and Parallel Sets [40] visualizations. These visualizations cleverly utilize the display space to show individual attribute information in a compact way, as well as provide interactivity to explore attribute relationship patterns.

Parallel Coordinates: A Parallel Coordinates visualization represents each attribute using a vertical axis and each data record as an x-monotone polygonal line or curve intersecting these axes according to the values present in the record (Figure 1.1(a)). For a numeric dataset, a Parallel Coordinates visualization may provide detailed information about individual attribute as well as their pairwise relationships. For example, one can examine the value distribution of an attribute by examining the intersection points along the vertical axis. The axes are usually reorderable and one can examine the potential relationships among two adjacent attributes by examining the visual pattern appearing between the corresponding vertical axes.

A dataset with categorical attributes poses a problem to Parallel Coordinates since the lines can overlap each other. For example, consider a dataset with three attributes (X, Y, Z), each consisting of two categories. A Parallel Coordinates visualization of such a dataset will create at most $2 \times 2 \times 2 = 8$ distinct polygonal lines (Figure 1.1(b)), even when the dataset contains hundreds of records.

Parallel Sets: The Parallel Sets visualization is especially designed for categorical data. Instead of showing individual data points, a Parallel Sets visualization reveals data frequencies by drawing parallelograms



(a) (b) (c)

Figure 1.1: Parallel Coordinates visualizations with (a) numerical attributes and (b) categorical attributes. (c) A Parallel Sets visualization.

between pairs of categories (Figure 1.1(c)). The width of the parallelogram is relative to the number of data records that share those categories. This idea gives rise to the notion of sets. For example, let (x_1, x_2), (y_1, y_2) and (z_1, z_2) be the categories for the attributes X, Y, Z we considered earlier. Then for each i, j, k, where $1 \leq i, j, k \leq 2$, a Parallel Sets visualization forms a set S_{ijk} by collecting the data records that belong to all three categories x_i, y_j and z_k. Assume that the axis X, Y, Z are horizontally aligned. Then the set S_{ijk} is represented using two parallelograms: one corresponding to S_{ij} and the other to S_{jk}. Here S_{ij} is a *predecessor* of S_{jk}. We refer to such a sequence of contiguous parallelograms as a *ribbon*. Based on the context, sometimes a pair of ribbons may be interpreted as one ribbon being partitioned into two ribbons. For example, in Figure 1.1(c), the two ribbons corresponding to S_{221} and S_{222} may be interpreted as the ribbon $S_{x_2y_2}$ being partitioned into ribbons $S_{y_2z_1}$ and $S_{y_2z_2}$.

If the attributes are numeric, then a Parallel Sets visualization can be designed by transforming each numeric attribute into a categorical attribute, where the categories are defined as value ranges [40]. For example, the numeric attribute body weight can be transformed into categories: Underweight, Healthy weight, Overweight and Obese.

1.1 Motivation

The booming computing resources, the popularity of social media, and the widespread use of technologies provide businesses, industries, and even individuals a boost in the ability to collect heterogeneous data. These data are typically large and complex with multiple attributes, which can be numerical or categorical. Multi-attribute data are hard to understand and analyze. Data visualization has been playing a big role in data analytics, which helps researchers to reveal patterns, focus on important information, and ignore disruptions. People are already familiar with and have mature techniques to process different types of homogeneous data into databases in specialized scenarios such as for multimedia datasets or medical records, but designing a standard practice for visual exploration of multi-dimensional data with numerical, categorical, and spatial attributes still has a long way to go. Researchers have been using various techniques to visualize multi-dimensional datasets and some of these work perfectly in specific situations, but there is yet no widely-adopted best practice to visualize such data. One of the main challenges for creating a standard visualization for mixed data is that due to the variety of attributes, they often require multiple views to be integrated harmoniously for effective exploration. The visualization that works for numerical data usually performs poorly for categorical data, and vice versa. The problem becomes even more difficult when the data records are associated with spatial and temporal information.

To work with mixed data types, one can investigate the numerical and categorical attributes separately using different information-linked views, but it is hard to build relationships between views and if not designed carefully, then it could be misleading and increase the effort to find useful information. To be more specific, different visualizations of a dataset are visualized across different views simultaneously and interactions (such as selection or filtering) in one view are reflected across multiple other views. This approach ensures the flexibility of different visualization techniques being used within its most suitable situations and also helps users to build relationships across views. However, an ill-designed visualization may significantly increase users' cognitive load, and mislead users with ambiguous or unclear information. This motivated us to examine if we can find a balance between information interpretation and design complexity.

1.2 Research Questions

The thesis aims at answering the following specific research questions:

RQ1. How can mixed data with numerical and categorical attributes be harmoniously visualized such that one can understand the following information?

A. Distribution of numeric attribute values and sizes of various sets determined by categorical attributes.

B. Statistical summaries of the attributes and spatial distribution of data records (e.g., when the data records are associated with spatial locations).

C. Potential pairwise relations among attributes, and possible extensions, i.e., their interactions over a spatial domain.

RQ2. How should the visualizations appear when analyzing real-life datasets in various applied domains?

RQ3. What are the performance bottlenecks while building such visualizations and how to cope with them?

RQ4. What are the design considerations one must take when creating visualizations for mixed data?

1.3 Contribution

We propose a visualization Set-Stat-Map for spatial data with mixed numeric and categorical attributes combining the idea of Parallel Sets, statistical columns, map-based interactions, and multiple coordinated view techniques (Chapter 3, **RQ1**). The Parallel Sets view displays the data records by grouping them into sets (displayed as ribbons), which provides information about the relationships among data attributes and the number of data records in each set. We rotate the Parallel Sets visualization (from vertically to horizontally) to better link the sets determined by the last axis with the rows of the statistics columns. Each row of the statistics columns provides statistical summaries of the numeric attributes for each category of the last axis of the Parallel Set. By selecting a row of the statistics columns, one can examine the data records on the map view. The interactive features (transformation of numeric attributes into categorical ranges, axis reordering, color and texture selection, filtering based on map selections, etc.) allow users to explore the dataset by generating visual configurations that best suit their needs.

We also proposed two enhancements for the Parallel Sets view. The original Parallel Sets visualization only uses colors for ribbons to help users quickly distinguish categories from the first axis. Our first enhancement adds textures relating to the second-axis categories to help users further analyze a pair of attribute combinations. Our second enhancement to the Parallel Sets view is merging unimportant sets into one in order to limit the number of possible set combinations, which provides better system performance and reduces the user's cognitive load.

We demonstrated the usability of Set-Stat-Map with use cases from different domains: scientific, commercial, and online forum. We chose a meteorological datasest for the scientific domain, an Airbnb dataset for the commercial domain, and a StackOverflow dataset for the online forum (Chapter 5, **RQ2**). We evaluated the performance of Set-Stat-Map to find out its limitations and gave our suggestions according to the system evaluation results (Chapter 5, **RQ3**). We show both the strengths and weaknesses of Set-Stat-Map when visualizing mixed data and provide guidelines to effectively use such a visualization (Chapter 6, **RQ4**).

When visualizing the meteorological dataset (Figure 1.2), users can easily find the yearly trend of a weather attribute in a specific area; also, users can easily compare two different atrributes in a specific area to see if they are related and how they relate to each other. When visualizing Airbnb dataset, users can quickly get which area is more popular, is more expensive to rent, or having better user ratings. When visualizing StackOverflow dataset, users can easily figure out which topics are discussed by more users, user retention for each technology over time, and posible relationships between topics. However, first-time users of Set-Stat-Map might need some time to learn the system to understand the visualizations and to be familiar with the interactions.

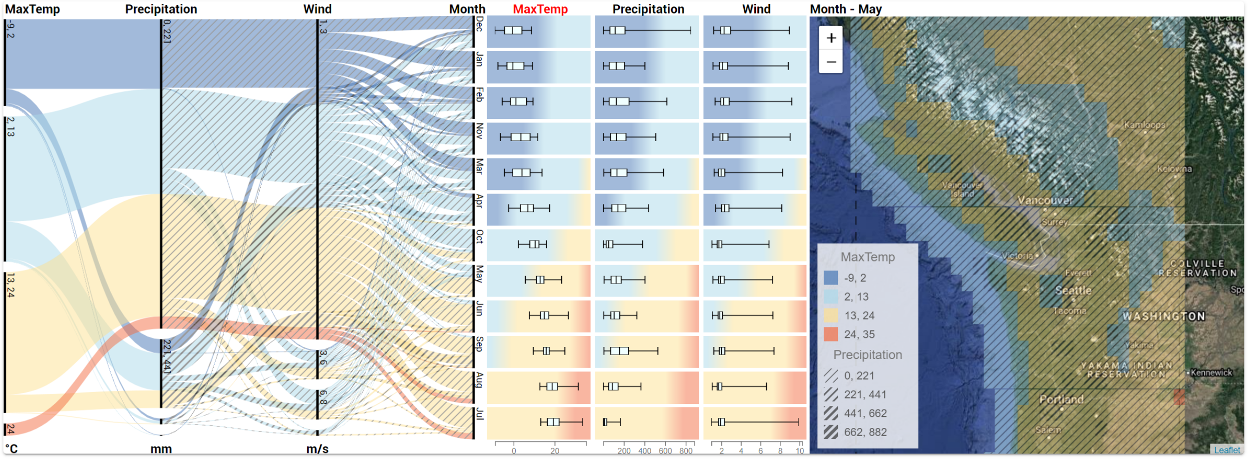


Figure 1.2: Set-Stat-Map visualization for a meteorological dataset of Vancouver-Seattle region. The variable maximum temperature, precipitation, and wind speed are selected. The month axis is sorted based on maximum temperature in ascending order. The month of May is selected to show on the map view.

1.4 Evaluation

An implementation of Set-Stat-Map is available at GitHub. We evaluate our system in the aspects of generalizability, performance, and usability. First, to evaluate the generalizability, we adapted our system to datasets from three different domains that contain a combination of numerical, categorical, and spatial data. Each dataset contains at least two of the three data types. We found that our system can generate visualizations for datasets from all three domains with minimal effort. Then we evaluate the performance by trying to use small and large datasets for each domain to test how the performance is affected by the dataset size. We also try to display different levels of details, which need to render a different number of view elements to find a balance between performance and user experience. Our evaluation suggests that our system can provide reasonable details with real-time user interactions to obtain insights into the data. Finally, we evaluate the usability by giving our system to experts, where we let them explore the system and ask them to explore and perform some tasks using our system. From their feedback, our system shows its value to help experts analyze complex multi-dimensional datasets in specific areas.

1.5 Chapter Organization

This thesis is organized into seven chapters, including the current chapter. In Chapter 2, we discuss some existing methods of visualizing multi-dimensional datasets, which we arrange by four different themes. We first present axis-based techniques that provide detailed and accurate information about each dimension in the datasets. We discuss traditional axis-based techniques could be challenging to explore when visualizing datasets with more than three dimensions, yet some newer techniques, such as Radar Chart and Parallel

Coordinates, can provide smart solutions for it. Then we discuss set-based techniques, such as Parallel Sets and Bubble Set, which group similar data records into sets to help reduce users' cognitive load. After that, we present table-based techniques. Table-based solutions can provide all detailed information at the cost of lacking intuitive graphical views. Finally, we discuss map-based techniques, which are useful for visualizing datasets that contain spatial attributes and are usually used within multi-view visualizations.

In Chapter 3, we discuss how we designed Set-Stat-Map. We first demonstrate its three main views. The first view leverages the idea of Parallel Sets, which combines multiple groups of similar data records into sets and presents them in a space-effective way. We also discuss some improvements and modifications of our version of Parallel Sets. The second view, Statistics Columns, visualizes a set of numerical attributes in a table-like manner, which has various heights that match the height of the last axis of the Parallel Set. The third view is the map view, which is used to demonstrate spatial information in the dataset and provides some user interactions that can affect the other two views. We then talk about how we make all three views work as a whole and how users can interact with the system. Finally, we discuss the design rationale that makes the multi-view visualization works.

In Chapter 4, we talk about how we process the data and implement the system. We first gather the raw data from different domains, which may be of any format and could contain information that we do not need. After gathering the raw data, we write scripts for each type of raw data to clean and restructure it and store it in an SQLite database file in a tabular manner. We then build our visualization for each domain based on the processed data format. For most cases, we combine the data processing and the data visualization into a single web-based solution, so that users only need a browser to access the whole system. Finally, we conduct development iterations for the meteorological dataset with feedback from experts in the meteorological domain, and for the other datasets with general participants.

In Chapter 5, we conduct the performance and usability evaluation of the system. To evaluate the performance, we first run multiple test cases and record the statistics in aspects of visual complexity and time complexity. We then plot the collected data to analyze the performance and give our suggestions. To evaluate the usability, we adapt the system to three different dataset types. The first one is a meteorological dataset in Canada, which contains information about temperature, humidity, solar radiation, etc. We generated datasets for British Columbia, Saskatchewan, and Ontario. The second one is from StackOverflow, which is a software developer community. It contains information about users' locations, topic, the year count of user's profile creation, etc. The third one is about Airbnb, a well-known online vacation rental service. The dataset contains price, feedback, location, etc. We load a dataset of Calgary into our system. For each use case, we discuss if it fits our visualization and how experts think it can help them to better understand the dataset.

In Chapter 6, we discuss how and why we choose to design our system in the current way. We point out some key factors of designing our visualization and discuss the pros and cons of different potential decisions that we could make for them. We also talk about the potential future works that may improve the system.

In Chapter 7, we summarize this thesis. It recalls the problem and the major contributions.

1.6 Declaration

Throughout this document, the term ”we” refers to the author and the reader following the theoretical computer science norm for scientific writing.

A preliminary set of the results has appeared in the proceedings of 2022 IEEE 15th Pacific Visualization Symposium (PacificVis) [59] and in the Journal of Visual Informatics [43].

2 Related Work

Previous research has explored different ways to visualize multi-dimensional datasets. Most traditional techniques rely on traditional charts, plots, or different coordinate systems to provide detailed attribute information for each record, some try to embrace the idea of sets in order to show the big picture of the dataset in a clearer and simpler way, and some try to apply multiple views as a whole, which overcomes some limitations of single-view visualizations. In this section we discuss the literature research into themes — axis-based, set-based, dimensionality reduction based, table based, map based, and multi-view based techniques.

2.1 Axis Based Techniques

Coordinate systems have been widely used to visualize two and three-dimensional data for years, which proved their value in visualizing accurate details for such data types. A coordinate system uses its axes to hold possible values of both numerical and categorical types, and it presents data records at the intersection of a set of attributes (one from each axis). In general, for k attributes, one can construct $\binom{k}{2}$ scatterplots to examine pairwise relations.

2.1.1 Extending Statistical Plots

For some simple cases, extending existing two-dimensional visualization techniques gives a straightforward solution. For example, a bar chart is commonly used to visualize two-dimensional datasets. One can apply different colors or textures to display an extra dimension. The discrete colors and different patterns of textures can be used to demonstrate categorical attributes; the gradient of colors and density of textures are sometimes used to map numerical attributes. Another example can be a scatter plot, which is also used to visualize two-dimensional data. However, one can use different shapes, colors, and sizes to make it to be able to display information for more dimensions.

2.1.2 Radar Chart

Radar chart [46, 50] extends the idea of coordinate systems, where each axis is put in a circular manner sharing the same origin point to overcome the limitation of the standard 2D or 3D Cartesian coordinate systems. Each data record is represented as a polygon with vertices lying on various axes depending on the record's value at that dimension. Overlaps of the polygons can show the similarities between different data records.

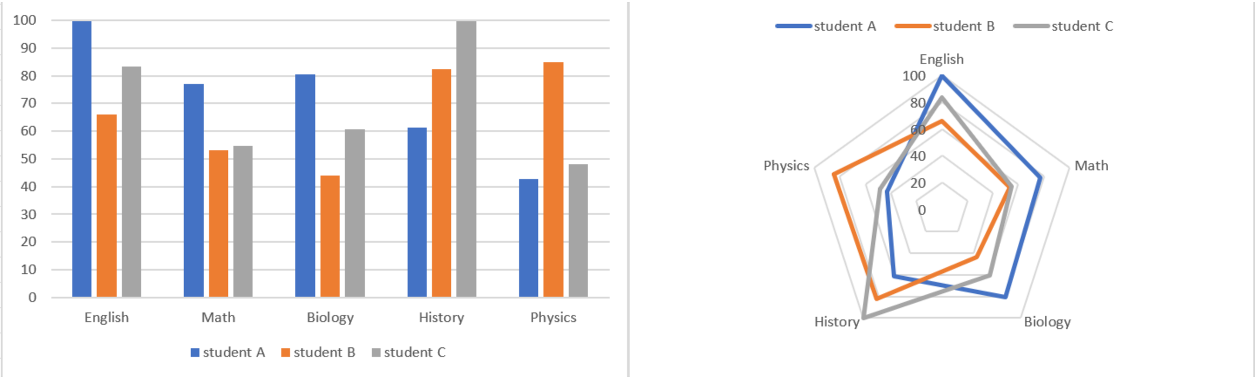


Figure 2.1: The same data of three students' academic performance is shown by bar chart (left) and by radar chart (right), respectively.

Figure 2.1 is an example of the same dataset being presented by bar chart and radar chart, respectively. In this example, the radar chart not only clearly shows the similarities between different students by their overlaps but also shows the values of each attribute by the intersections on its axes. However, when the number of data records gets larger, it fails because too many overlaps make the visualization indistinguishable.

2.1.3 Parallel Coordinates

In a Parallel Coordinates visualization [39], axes are placed parallel to each other, where each data record is presented as a polygonal line. Compared to radar charts, a Parallel Coordinates visualization more efficiently utilizes the space, i.e., it provides uniform space (instead of a cone-shaped region) between a pair of adjacent axes, and the line patterns between adjacent axes are easily comparable since they are parallel to each other (instead of being arbitrarily rotated). Line overplotting (when the number of data records is large) and axis ordering (choosing a linear ordering that represents the most interesting pairwise relations) are two common problems in a Parallel Coordinates visualization. A number of research works attempted to leverage Parallel Coordinate to effectively visualize categorical attributes [49, 62]. We refer interested readers to [37] for a survey on various extensions of Parallel Coordinates visualization. Figure 2.2 (left) shows a Parallel Coordinates visualization with six axes.

2.1.4 Box Plot

Box plot [61, 47] is a graphical representation that gives information about the distribution of data. It could look slightly different in various implementations, but they usually contain the following information in common: the minimum, the first quartile, the median , the third quartile, and the maximum of the dataset. Figure 2.3 (left) shows an example of box plot that shows distribution of student' grades for two classes. In this example, the filled box area stands for the range from the first quartile to the third quartile. The horizontal line inside the box stands for the median of its corresponding data. The two separate horizontal lines at the top and at the bottom, which are connected with vertical lines, stands for the minimum and

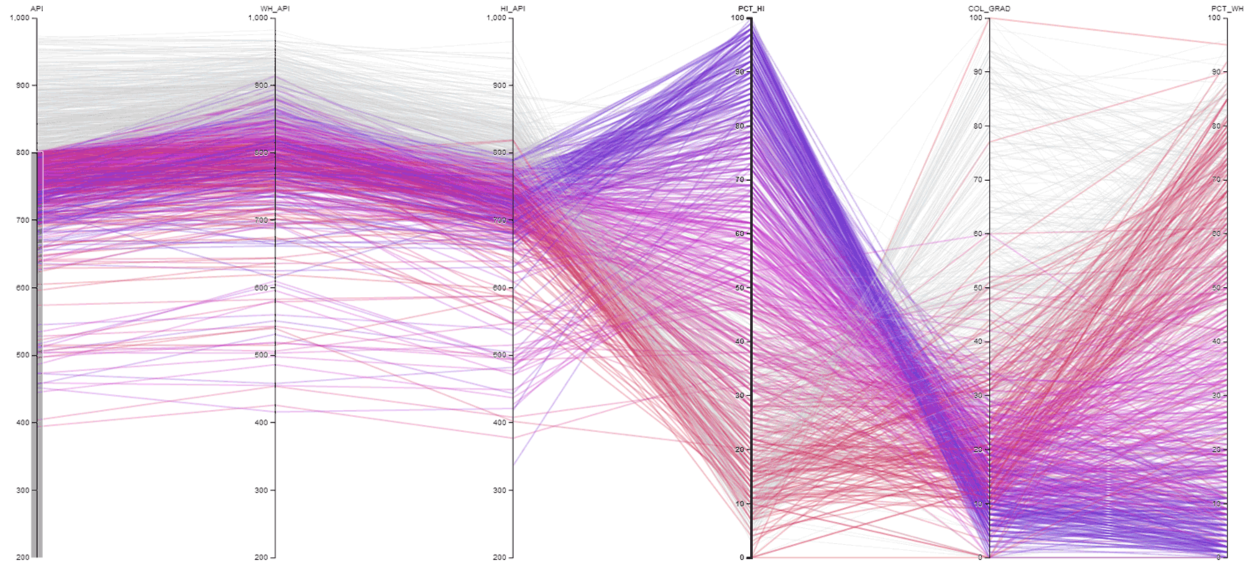


Figure 2.2: Parallel coordinates visualization [6].

.

maximum value of the data, respectively.

2.2 Set Based Techniques

It is also important to quickly understand the big picture of a large-scale multi-dimensional dataset, such as the similarities and relationships of data records across different attributes. To achieve such a goal, researchers tried to embrace the idea of sets in multi-dimensional data visualizations. Set based techniques for visualizing multi-attribute data construct sets based on various-category combinations[18].

2.2.1 Venn Diagram

The most common and traditional set visualization technique is Venn Diagram [22, 32]. It uses multiple ellipses with overlaps to show different sets and their intersections. By extending the idea, the shapes it contains can be any complex polygons [32, 55]. Figure 2.3 (right) demonstrates a Venn Diagram with six sets using complex polygons. The area of those shapes can demonstrate the size of sets and intersects, and sometimes colours are added as well to make sets more distinguishable. Even though we can use complex polygons, it becomes unclear when the number of sets gets larger.

2.2.2 Parallel Sets

Parallel Sets [40] is a good example of using sets to visualize multi-dimensional datasets, where it is possible to define a set for any category combination that contains one category from each attribute. Since Parallel Sets visualization use parallelograms to connect adjacent axes, the visualization quickly becomes occluded as

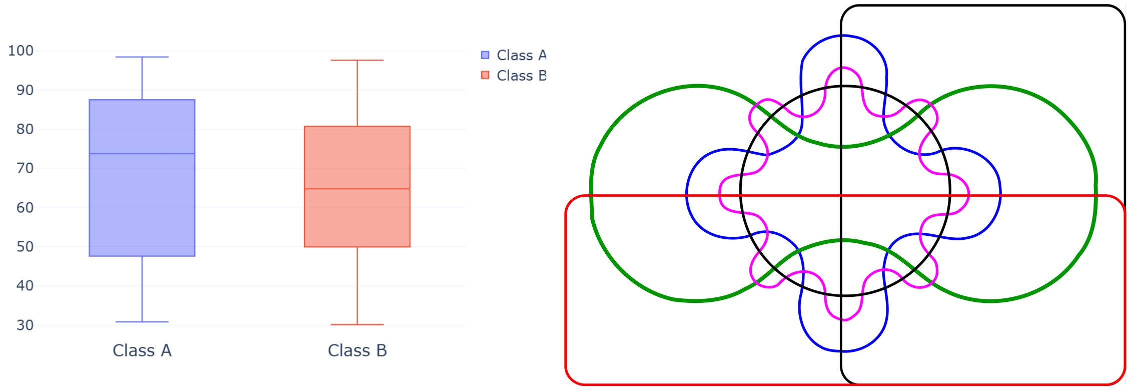


Figure 2.3: Box plot showing the distribution of students' grades for two classes (left); Venn Diagram visualization with six sets (right) [13].

.

the number of categories grows. Therefore, Parallel Sets are most suited for categorical attributes with only a few categories. Figure 2.4 shows an example of Titanic survivors visualized with the Parallel Sets technique.

2.2.3 Bubble Sets

Spatial data are often visualized on maps. Bubble Sets [23] is an example that visualizes contours (implicit surface) for every category (set) that encloses all its elements and excludes all other elements if possible. Thus multiple category combinations can be visualized directly on a map showing how the elements of each category are spatially distributed. However, such an approach is not as suitable as Parallel Sets in revealing frequency information and relationships among two categorical attributes. Region based set visualization approach [17, 16] may create cluttered visualization only for a few categories for irregular and meandering regions. Figure 2.5 demonstrates how one can overlay Bubble Sets on the map.

2.3 Dimensionality Reduction

Dimensionality reduction [58] is a technique that transforms a multi-dimensional dataset from a high-dimensional space to a low-dimensional representation, which should still maintain similar distances or map similarities to proximities among data points. After doing dimensionality reduction, the axis becomes abstract that does not stand for any specific dimensional information of the original dataset anymore. Thus, even though it successfully helps people to understand similarities between data records, it fails when people want to also understand the details of each attribute. There are multiple ways to do dimensionality reduction and different algorithms can demonstrate different kinds of similarities.

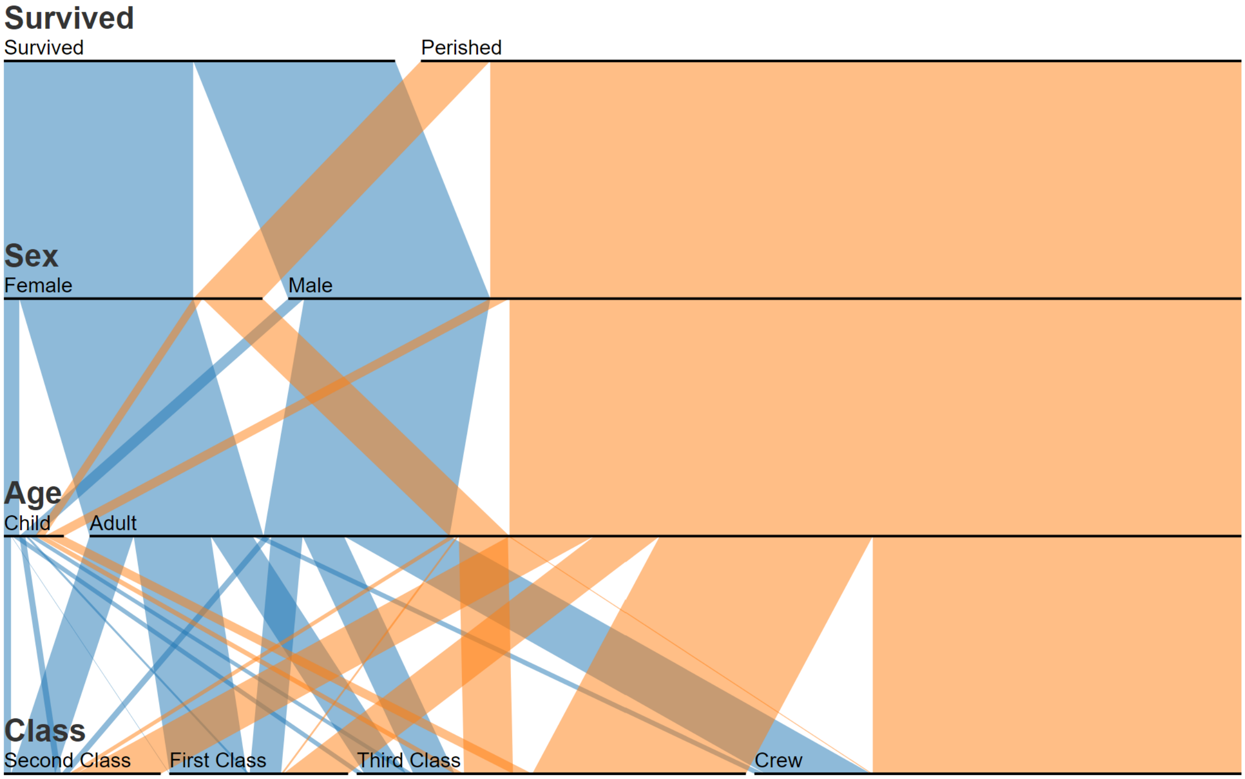


Figure 2.4: Visualizing Titanic survivors using Parallel Sets. [7]

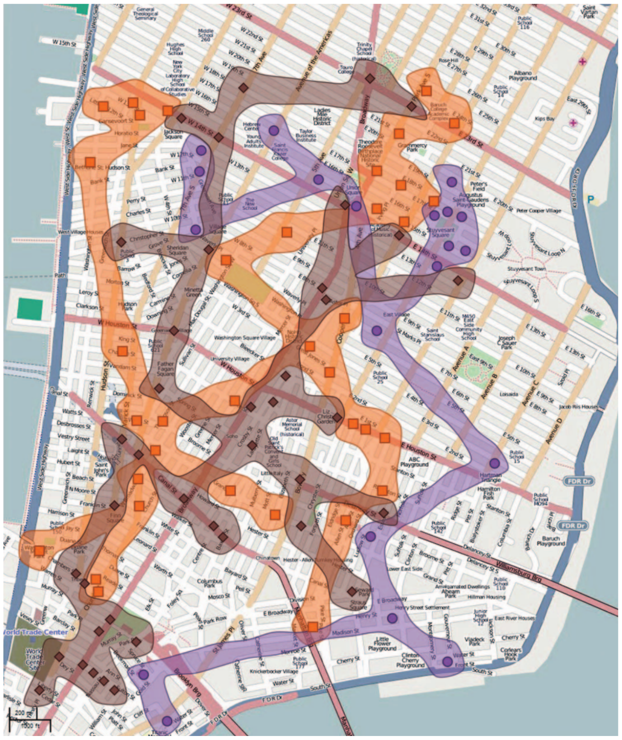


Figure 2.5: Bubble Sets overlays on a map showing hotels (orange), subway stations (brown), and medical clinics (purple) in lower Manhattan. [23]

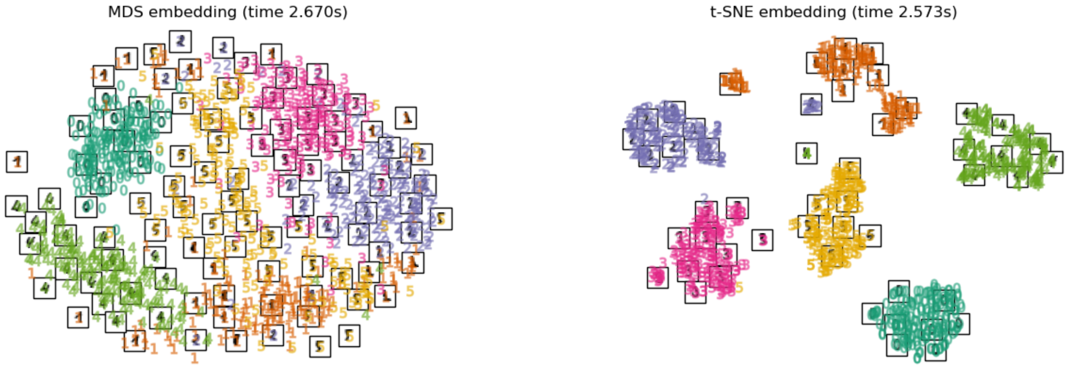


Figure 2.6: Plotting MDS (left) vs. t-SNE (right) for a digits dataset [5]

.

2.3.1 Multi-dimensional Scaling

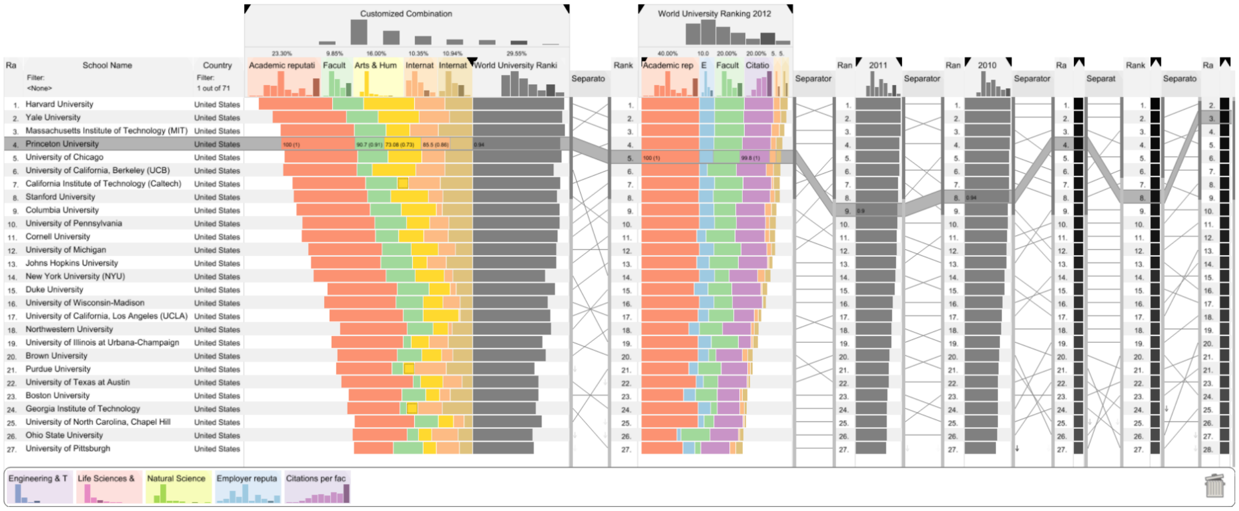
Multi-dimensional scaling (MDS) [24, 38] is widely used to visualize the similarity of data records in a multi-dimensional dataset. It extracts the distance information between data records from the multi-dimensional dataset and puts it into a lower-dimension Cartesian space. When using scatter plots, the less distance between two points means a higher similarity between these two data records. Figure 2.6 (left) shows a digits dataset plotted using MDS.

2.3.2 Principal component analysis

Principal component analysis (PCA) [15, 25] is another main method for dimensionality reduction, which is similar to MDS. The major difference between MDS and PCA is the distance measurement. Instead of focusing on the Euclidean distance of points like MDS, PCA focuses on the covariance of data and the result would not preserve the distance between points. It would be the same compared to MDS if the covariance of data and the Euclidean distance between data points are the same.

2.3.3 t-Distributed Stochastic Neighbor Embedding

The t-distributed Stochastic Neighbor Embedding (t-SNE) [57, 25] is a variant of Stochastic Neighbor Embedding to embed high-dimensional data into low-dimensional space with clusters for similar data points. It first defines a probability distribution according to the Euclidean distances of points from the high dimension, which stands for the similarity of the points. Then it gets probabilities for the low dimension, with a heavy-tailed distribution. With these two probability distributions, we can get the joint probabilities that minimize the mismatch between the high dimensions and the low dimensions. As a result, we get clearer clusters of similar data points. Figure 2.6 (right) shows a digits dataset plotted using t-SNE.



Ra	School Name	Country
	Filter: <None>	Filter: 1 out of 71
1.	Harvard University	United States
2.	Yale University	United States
3.	Massachusetts Institute of Technology (MIT)	United States
4.	Princeton University	United States
5.	University of Chicago	United States
6.	University of California, Berkeley (UCB)	United States
7.	California Institute of Technology (Caltech)	United States
8.	Stanford University	United States
9.	Columbia University	United States
10.	University of Pennsylvania	United States
11.	Cornell University	United States
12.	University of Michigan	United States
13.	Johns Hopkins University	United States
14.	New York University (NYU)	United States
15.	Duke University	United States
16.	University of Wisconsin-Madison	United States
17.	University of California, Los Angeles (UCLA)	United States
18.	Northwestern University	United States
19.	University of Illinois at Urbana-Champaign	United States
20.	Brown University	United States
21.	Purdue University	United States
22.	University of Texas at Austin	United States
23.	Boston University	United States
24.	Georgia Institute of Technology	United States
25.	University of North Carolina, Chapel Hill	United States
26.	Ohio State University	United States
27.	University of Pittsburgh	United States

Figure 2.7: LineUp shows a ranking of the top universities in a 2012 ranking. [33]

2.4 Table Based Techniques

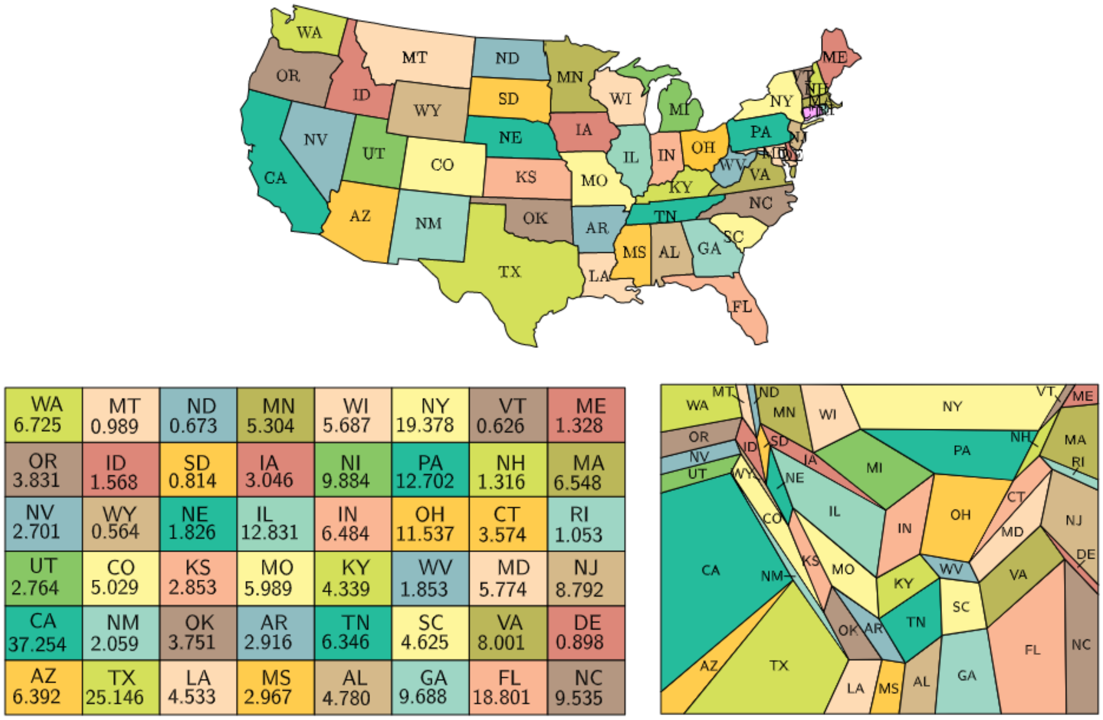
The spreadsheet provides a tabular visualization for the multi-attribute dataset, where the rows correspond to the data records, columns correspond to the attributes, and each table entry is either a value or a category of its column attribute. If the precise numerical values are not important, then replacing the numerical values using horizontal bars and sorting the rows by the attribute of interest gives a quick overview of the value distribution. However, sometimes tabular visualizations appear to be sparse and are not scalable with increasing data size.

2.4.1 LineUp

LineUp [33] takes a step further and allows users to examine the ranks of data records, where a rank of a record is determined by a weighted sum of its attribute values. To improve the scalability of the visualization, LineUp allows users to collapse columns, and shows only the top 100 rows at a time whereas the rest of the rows are shown in a compressed manner which can be explored using fish-eye selection [52]. Figure 2.7 shows the main view of the LineUp visualization.

2.4.2 Table Cartogram

A table cartograms [31, 34, 30] is a weighted two-dimensional table that has transformed cell shapes so that each area of its cell represents its weight. With the shape transform, it still somehow maintains the same adjacency and relative positions. It can be a good choice to visualize tabular data with weights, but it can get too skewed in some cases, which dramatically increases users' cognitive load. Figure 2.8 shows a table cartogram comparing to the geographical map and table.



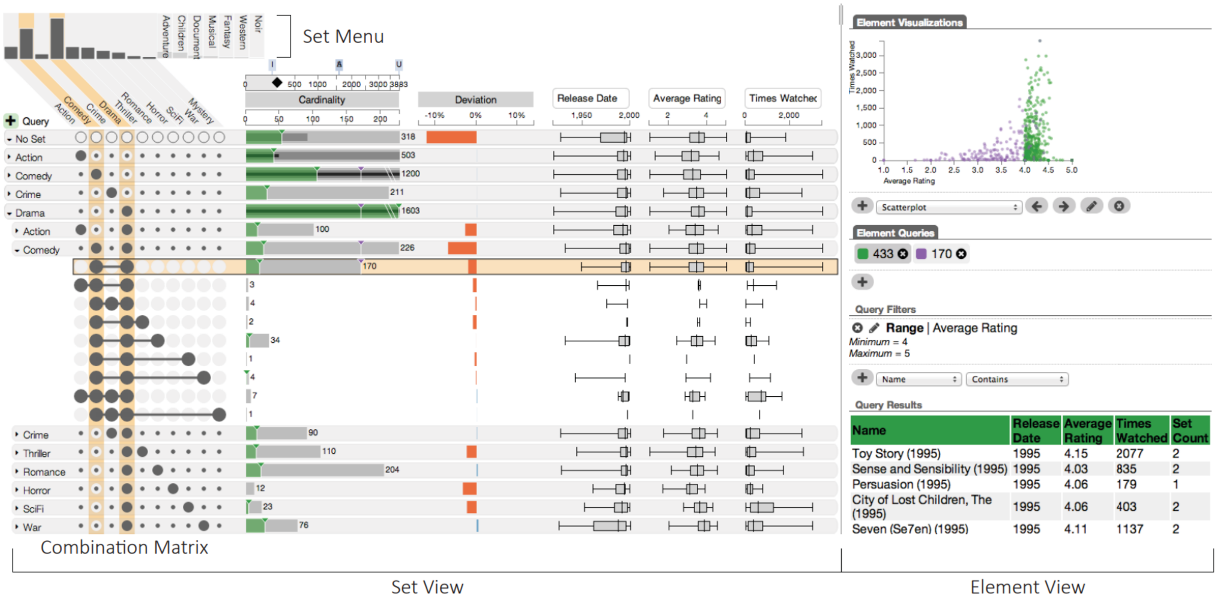
WA 6.725	MT 0.989	ND 0.673	MN 5.304	WI 5.687	NY 19.378	VT 0.626	ME 1.328
OR 3.831	ID 1.568	SD 0.814	IA 3.046	NI 9.884	PA 12.702	NH 1.316	MA 6.548
NV 2.701	WY 0.564	NE 1.826	IL 12.831	IN 6.484	OH 11.537	CT 3.574	RI 1.053
UT 2.764	CO 5.029	KS 2.853	MO 5.989	KY 4.339	WV 1.853	MD 5.774	NJ 8.792
CA 37.254	NM 2.059	OK 3.751	AR 2.916	TN 6.346	SC 4.625	VA 8.001	DE 0.898
AZ 6.392	TX 25.146	LA 4.533	MS 2.967	AL 4.780	GA 9.688	FL 18.801	NC 9.535

Figure 2.8: The map (top), the grid (lower left), and the table cartogram (lower right) visualizations for the population of US for 2010. [13]

2.5 Map Based Techniques

Choropleth maps and proportional symbol maps [28] are commonly used methods to visualize spatial information on a map. Although they are well-suited for uni-variate data, color blending and texture patterns (for choropleth), and glyphs (for symbols) can be leveraged to extend these techniques for bivariate visualization. Color blending is widely used in spatial analysis to find probable correlations between two geospatial variables [29] and researchers have examined different ways to construct trivariate choropleth maps [53, 54] and even 4-variate [63] maps.

Overlaying glyphs (icons, pie charts, and even complex textures) on a map is a popular way to visualize multi-attribute geospatial information [21]. Typically, glyphs are designed to encode data into features that can be perceived through preattentive visual channels [60]. Pexels, introduced by Healey and Enns [35], encode multi-attribute datasets into perceptual textures with height, density, and regularity properties. In general, the texture of varied contrast, density, and directionality [56, 42] are often used to visualize multivariate data.



Name	Release Date	Average Rating	Times Watched	Set Count
Toy Story (1995)	1995	4.15	2077	2
Sense and Sensibility (1995)	1995	4.03	835	2
Persuasion (1995)	1995	4.06	179	1
City of Lost Children, The (1995)	1995	4.06	403	2
Seven (Se7en) (1995)	1995	4.11	1137	2

Figure 2.9: Visualizing relationships of movie genres using UpSet. [41]

2.6 Multi-view Based Techniques

Even though the visualizations above described perform very well in their suitable cases, single-view visualizations are still strictly limited. To provide even more information in a single visualization, researchers explored the possibility of interactive multi-view visualizations. Nowadays, the use of multiple coordinated views is becoming increasingly common in heterogeneous data analytics, e.g., environmental data exploration [20, 14] and multivariate network visualization [43]. When users interact in one view, other views can be updated based on their interactions. In such a visualization, every single view can deal with a specific job very well and the limitation of a single view can be overcome by leveraging other views. In the following we give a few examples of multi-view based visualization techniques.

2.6.1 UpSet

UpSet [41] is a good example of a set visualization approach using multiple views, which can be used to examine complex multi-attribute datasets such as movie datasets. The user can construct a query using a visual query builder, e.g., visualize movies that must include the category adventure, but not comedy. UpSet first constructs all possible category combinations (sets) that satisfy the query and then uses those as rows. The statistical summaries (e.g., average rating, release date) of each set is then visualized using separate statistics column. UpSet supports multiple views, which are collapsible. For example, the users can choose to visualize query records in a scatter plot or simply as a data table. A number of matrix and network based visualizations [18] have been explored to further improve the scalability of set visualizations. Figure 2.9 shows how UpSet visualizes a movie genres dataset.

2.6.2 Geo-Coordinated Parallel Coordinates

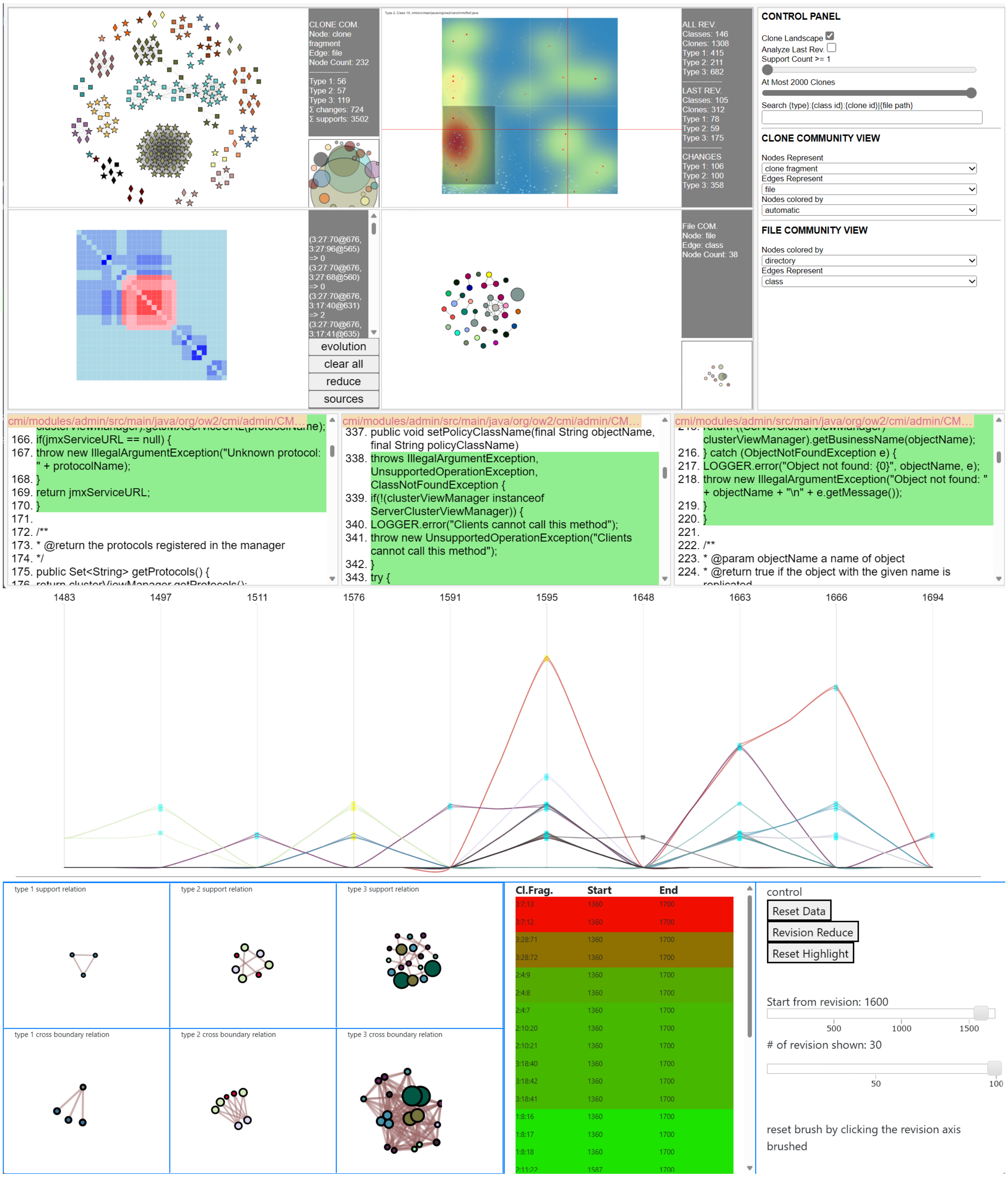
A geographic map, coordinated with charts or a Parallel Coordinates view, is another example of taking advantage of multiple views, where users can select a region of a map and examine the updated charts for the selected dataset. Since multiple coordinated views can visualize different aspects of the same dataset simultaneously, users can better comprehend the data. Such visualizations often lack methodological design efforts and the final visualization becomes a collection of charts, plots, tables, and maps that do not look like an integral part of the overall visualization and may immensely increase users' cognitive load. We refer the readers to [19, 48] for a rich body of research that examines design principals and usability issues of multiple coordinated views.

2.6.3 Clone World

Software clone [44] means the repeated or nearly similar code fragments existing in a software system, which is extremely common because it is highly likely being produced by copying and pasting code fragments during software development. In order to understand software clones in a software system, one needs to understand multiple aspects, such as relations between clones, relations between files, the evolution of clones, etc. As a part of our preliminary investigation of challenging datasets, we designed Clone World [43], which is a multiple interactive view software clone visualization to help people understand clones in a software system. Clone World provides a large number of views, and these views are highly connected. People can search, filter, and explore a software clone dataset in a single visualization. Figure 2.10 are the two main views of Clone World.

2.7 Summary

There are already many techniques for visualizing multi-dimensional and spatial datasets. Some of them add additional elements or mutate the visual arrangements for axis-based visualization techniques to be able to visualize multi-dimensional datasets, such as radar charts and Parallel Coordinates. Another way to visualize multi-dimensional datasets is to group the data into sets to focus on the big picture of a dataset. Examples of this type of technique are Venn Diagrams, Parallel Sets, and Bubble Sets. Dimensionality reduction is an efficient way to hide redundant information and highlight data points' similarities for multi-dimensional datasets, which creates clusters that converge similar points together and are usually plotted as scatter plots for visualization. MDS, PCA, and t-SNE are widely used algorithms for dimensionality reduction. Tables are natively used to visualize multi-dimensional datasets, which can be extremely helpful especially with some tweaks. LineUp and table cartograms are two examples of tabular visualizations. Choropleth maps and proportional symbol maps are commonly used for visualizing spatial information on a geographical map but they are not designed to handle multi-dimensional datasets. Interactive multi-view visualizations take advantage of different types of visualization and merge them harmoniously into a single system. UpSet, Geographic map coordinated with charts, and Clone World are some examples of multi-view visualization techniques. These existing techniques are good for their specific cases but perform badly when visualizing multi-dimensional datasets with mixed data types (numerical, categorical, and spatial). Using interactive multi-view techniques with other multi-dimensional visualization techniques could become a potential solution to the challenge of working with large mixed-type multi-dimensional datasets.



CLONE COM.
Node: clone fragment
Edge: file
Node Count: 232

Type 1: 56
Type 2: 57
Type 3: 119
Σ changes: 724
Σ supports: 3502

ALL REV.
Classes: 146
Clones: 1308
Type 1: 415
Type 2: 211
Type 3: 682

LAST REV.
Classes: 105
Clones: 312
Type 1: 78
Type 2: 59
Type 3: 175

CHANGES
Type 1: 106
Type 2: 100
Type 3: 358

(3:27:70@676, 3:27:96@565) => 0
(3:27:70@676, 3:27:68@560) => 0
(3:27:70@676, 3:17:40@631) => 2
(3:27:70@676, 3:17:41@635)

File COM.
Node: file
Edge: class
Node Count: 38

CONTROL PANEL

Clone Landscape ☑
Analyze Last Rev. ☐
Support Count >= 1
At Most 2000 Clones
Search {type}:{class id}:{clone id}|{file path}

CLONE COMMUNITY VIEW

Nodes Represent: clone fragment
Edges Represent: file
Nodes colored by: automatic

FILE COMMUNITY VIEW

Nodes colored by: directory
Edges Represent: class

cmi/modules/admin/src/main/java/org/ow2/cmi/admin/CM…

```
if(jmxServiceURL == null) {
throw new IllegalArgumentException("Unknown protocol: " + protocolName);
}
return jmxServiceURL;
}

/**
* @return the protocols registered in the manager
*/
public Set<String> getProtocols() {
```

cmi/modules/admin/src/main/java/org/ow2/cmi/admin/CM…

```
public void setPolicyClassName(final String objectName, final String policyClassName)
throws IllegalArgumentException, UnsupportedOperationException, ClassNotFoundException {
if(!(clusterViewManager instanceof ServerClusterViewManager)) {
LOGGER.error("Clients cannot call this method");
throw new UnsupportedOperationException("Clients cannot call this method");
}
try {
```

cmi/modules/admin/src/main/java/org/ow2/cmi/admin/CM…

```
clusterViewManager).getBusinessName(objectName);
} catch (ObjectNotFoundException e) {
LOGGER.error("Object not found: {0}", objectName, e);
throw new IllegalArgumentException("Object not found: " + objectName + "\n" + e.getMessage());
}
}

/**
* @param objectName a name of object
* @return true if the object with the given name is
```

Cl.Frag.	Start	End
3:7:13	1360	1700
3:7:12	1360	1700
3:28:71	1360	1700
3:28:72	1360	1700
2:4:9	1360	1700
2:4:8	1360	1700
2:4:7	1360	1700
2:10:20	1360	1700
2:10:21	1360	1700
3:18:40	1360	1700
3:18:42	1360	1700
3:18:41	1360	1700
1:8:16	1360	1700
1:8:17	1360	1700
1:8:18	1360	1700
2:11:22	1587	1700

Start from revision: 1600

# of revision shown: 30

reset brush by clicking the revision axis
brushed

Figure 2.10: The main view (top) and clone evolution view (bottom) of Clone World.

3 Design of Set-Stat-Map

The Set-Stat-Map visualization is partitioned into three views (Figure 3.1): Parallel Sets (left), statistics columns (middle) and map view (right). We first describe the details of each view and then the coordination among these views.

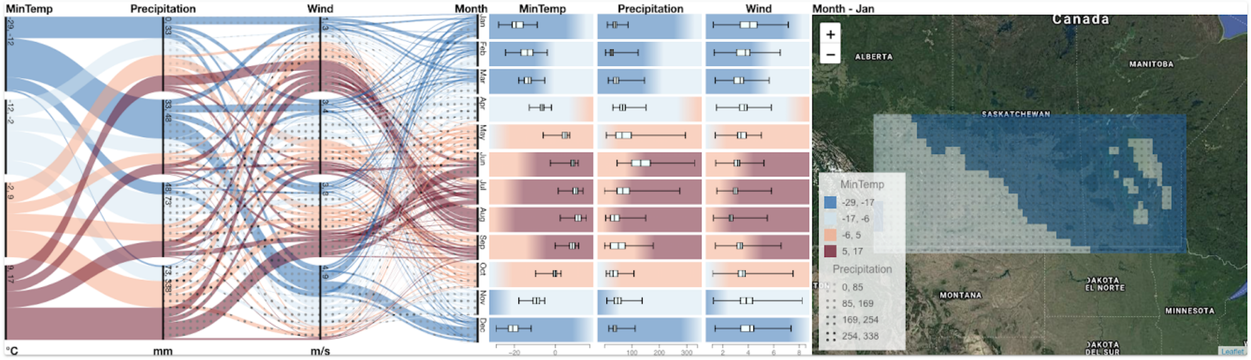


Figure 3.1: Three views of a Set-Stat-Map visualization: Parallel Sets (left), statistics columns (middle) and map view (right)

3.1 Parallel Sets View

The visualization constructs a Parallel Sets view with vertical axes. The first, second and the last axes (when ordered from left to right) are of most importance in our visualization.

Transforming numerical attributes to categorical attributes: The original Parallel Sets [40] showed how value ranges can be used to adopt numeric attributes in a Parallel Sets visualization by allowing users to specify value ranges of interest. We examine two additional options to automatically create such value ranges: one is partitioning the range of the attribute into some equal parts, and the other is to partition the range using quartiles. Both these two options (Figure 3.2) can be useful in practice.

Limiting the number of categories: A large number of categories for each attribute may create a large number of ribbons in the original Parallel Sets visualization [40], which dramatically reduces the width of the ribbons. It not only consumes expensive resources for rendering, but also makes some ribbons hard or even impossible to see. To improve this situation, we apply a configurable 'maximum category' limitation

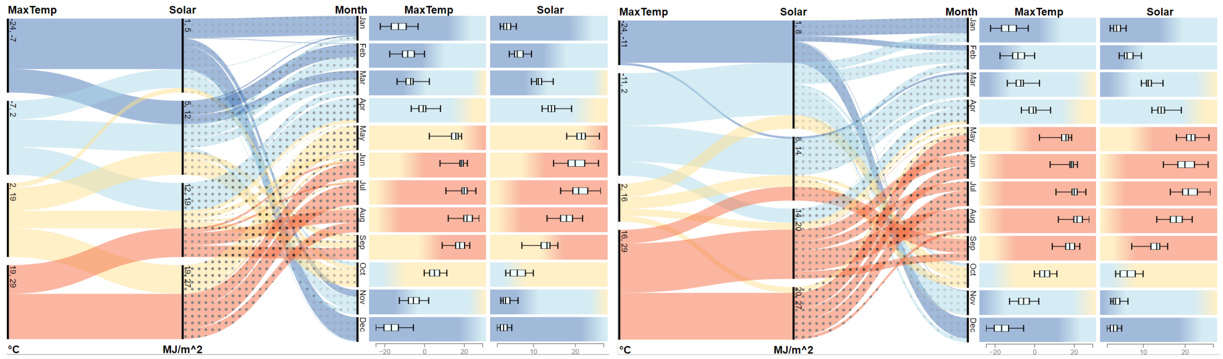


Figure 3.2: Users can explore two SET-STAT-MAP design by placing them one top of each other. For example, the left view shows an attribute transformation with quartile ranges, and the right view uses eight equal bins.

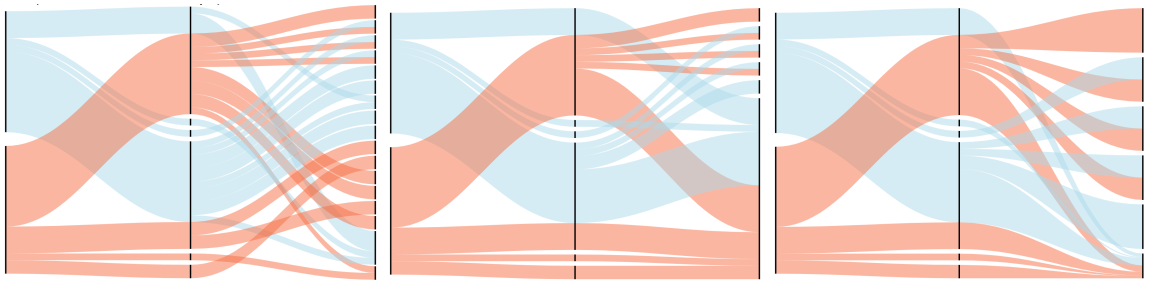


Figure 3.3: Parallel Sets (left) without any merged set, (middle) with merged set, and (right) with resized merged set. The number of categories at the last axis is limited to six, where the bottommost category is the merged one.

for each axis and merge the remaining categories. This greatly reduces the number of ribbons that need to be rendered. Figure 3.3 illustrates such an example.

In a default setting, we limit the number of categories per attribute to be displayed to a small number $\alpha \leq 8$. The rest of the categories are merged together and labelled as ‘other’. We set a small fixed width for the merged section and reshape related ribbons, which greatly improves the available space for the categories that have not been merged. Figure 3.3 (middle) and (right) show the difference between with and without limitation of the merged section size.

Since not all the categories are now visible, an obvious remedy is to filter the categories based on users’ needs, however, it is also possible to scroll at an axis to add more categories from the bottom end (‘other’ section) whereas the categories from the top join the ‘other’ section.

In a worst-case scenario, this may require rendering α^k ribbons, where k is the number of attributes. Hence the performance (rendering speed and interactivity) may decrease significantly for five or more attributes depending on the data. If more attributes are needed to be displayed, one can define a smaller value for α (even can customize the parameter for each axis).

Color encoding for the ribbons: We color a ribbon to represent its category on the first axis, which follows the original Parallel Sets design [40]. More specifically, we assign each category on the first axis using a distinct color. One can choose a color map based on the categories (qualitative color map for discrete options, or diverging color maps for numeric ranges). Let σ be the ordering of the colors from top to bottom. We then color each ribbon based on which category it belongs to on the first axis. For a category (for any attribute), there exists a vertical line interval intersected by a number of ribbons. The ribbons are ordered based on σ such that while examining such an interval, users can quickly recall where these ribbons are located at the first axis.

Texture encoding for the ribbons: We augment the original Parallel Sets [40] by adding texture to the ribbons. The textures are rendered on the ribbons from the second axis. We assign each category on the second axis using a distinct texture pattern. We then render a texture on each ribbon based on the category it belongs to on the second axis. We allowed different texture patterns and they usually fit into one of three types.

The first type uses arbitrarily chosen shapes for textures. For this type, the patterns can be circles, crosses, lines, triangles, etc. It works great for categorical axes in a dataset that creates just a few ribbons and each ribbon has enough vertical space to display the whole pattern, but, due to the complexity of these patterns, it would become messy and indistinguishable when ribbons become more narrow. Figure 3.4 (left) shows an example of the first type of texture pattern.

The second type maintains the same shape for all textures to provide a more clean and intuitive visualization for numerical value based axes. Figure 3.5 shows three examples of this type of texture. The top one uses dots for all textures but the opacity levels become larger when the value is larger; the middle one also uses dots but their sizes are used to distinguish the value level; the bottom one uses lines with various widths for different value levels. This type of pattern is likely to work best when the number of segments in each axis is bounded by four, or we only care more about knowing which value is larger, because we cannot easily distinguish the specific group of a texture pattern when more segments overlap and textures become very similar.

The third type focuses more on distinctiveness between multiple textures while still keeping some clues of value comparison. Figure 3.6 demonstrate a list of textures for this type, which combines shape difference (nothing, dot, and lines), width difference, and angle difference. The rationale behind allowing the second one (line-orientation pattern) is that for numeric ranges, the higher value ranges will naturally correspond to a larger angle of inclination of the lines with the x-axis. If the number of categories is at most four, then we use the first, third, fifth, and seventh patterns. Even when the categories correspond to some discrete non-numeric options, this natural texture ordering is likely to allow users to easily recall their top to bottom ordering on the second axis. Figure 3.4 compares the first type (left) and the third type (right).

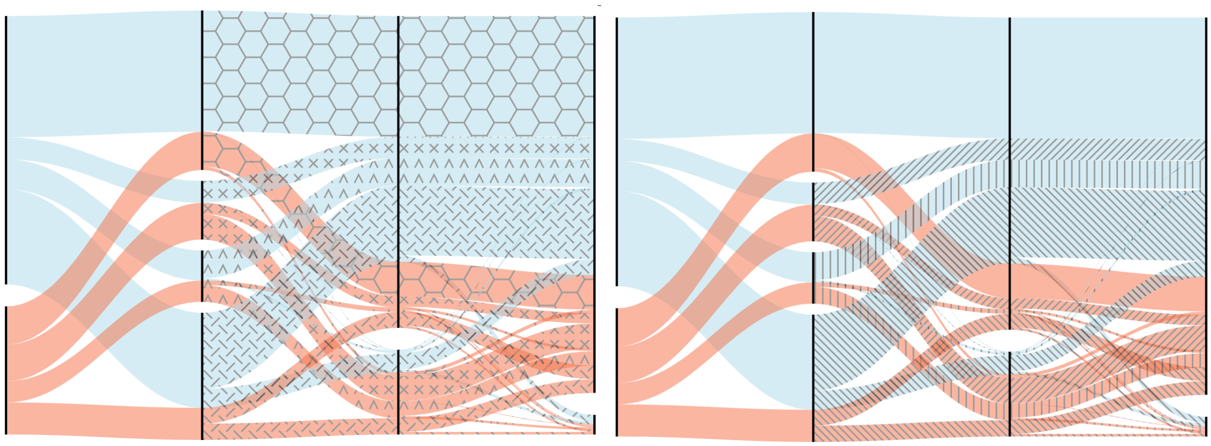


Figure 3.4: Parallel Sets (left) with arbitrary textures, and (right) with line-orientation based textures.

3.2 Statistics Columns View

The statistics column view consists of a set of numerical attributes. Each attribute is visualized in a column, where the rows are determined by the last axis of the Parallel Sets design. The jth row in a column (i.e., a cell) can be a box plot or a bar representing the summary statistic of the jth category of the last axis of the Parallel Sets. The box plot gives information about the quartiles and data range, and the bar plot shows the average value of the set (determined by the jth category) for the corresponding attribute. Figure 3.7 shows an example of statistics columns with box plots for each cell.

The background of a cell is filled using a colour gradient using the colors present in the jth category. Let I_j be the vertical interval corresponding to the jth category. Then the width of a color range in the jth row is determined by the proportion of interval taken by the corresponding ribbon at I_j. We use the same color order as it appears on I_j and set the height of a cell to be equal to the length of I_j.

The rationale behind coloring the background of the cells is to provide users with the composition details for the categories of the first axis in the Parallel Sets design. Sometimes, the interval I_j may be small and thus the ribbons may be hard to inspect. The gradient fill leverages the cell space to ease such inspection. The uniform cell space and linear ordering of the color gradients make it easier to compare various cell compositions visually with each other.

Figure 3.8 shows the statistics columns for the Saskatchewan weather dataset for three different sorting methods. Each column has its rows aligned to the last axis (Month) of the Parallel Sets view. The horizontal color gradient of each row are showing the same information about ribbons connecting to each segment of the last axis. The colours on segments of vertical axis are relatively hard to distinguish due to the limited height (even worse when each segment has a different height). which becomes easy to understand when showing horizontally on rows of statistics columns with a wider fixed space. When we click on a statistics column header, it toggles unchanged/ascending/descending sorting method on the Parallel Sets view's last axis for

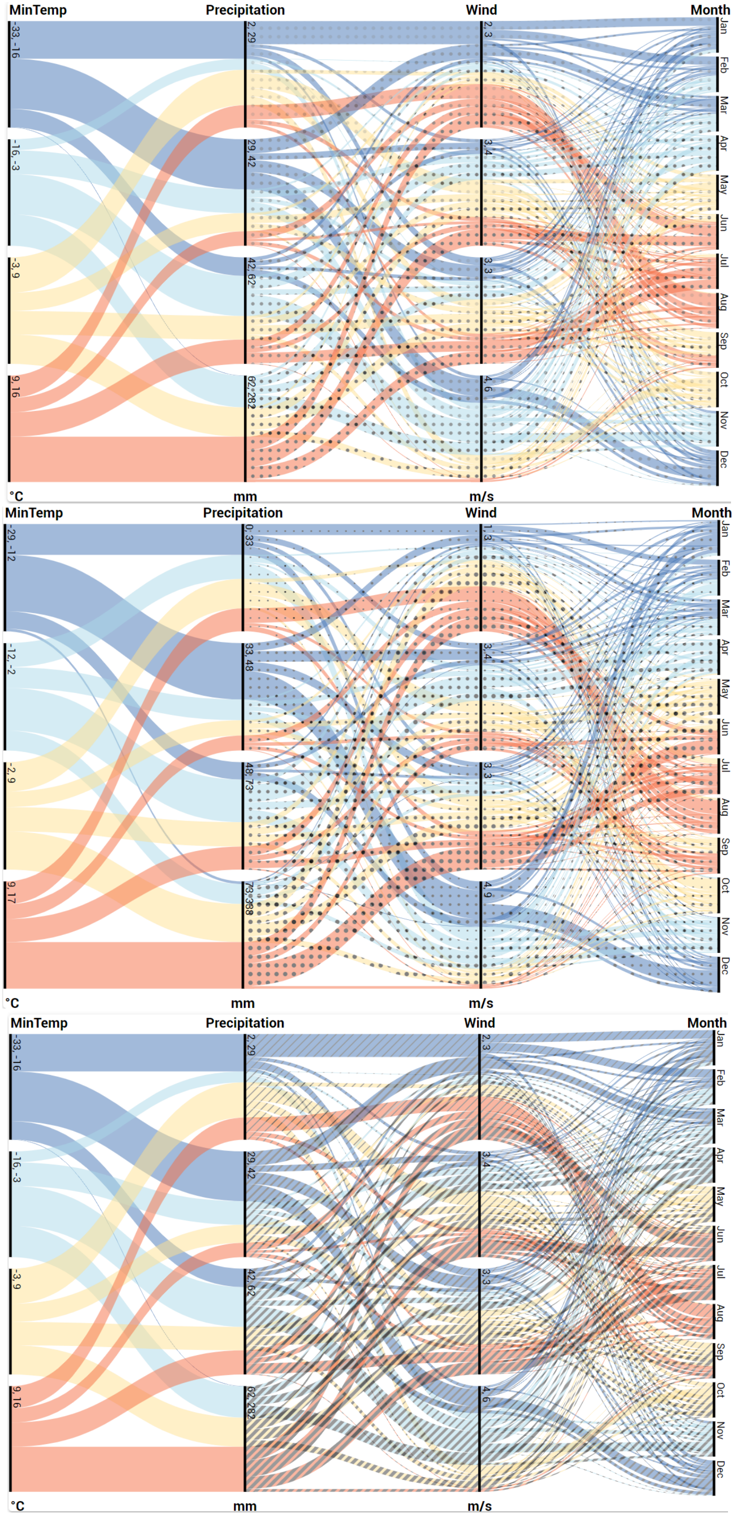


Figure 3.5: Three different ribbon textures for numerical value grouped axes: dots with different opacity levels (top), dots with different radius (middle), and lines with different widths (bottom).

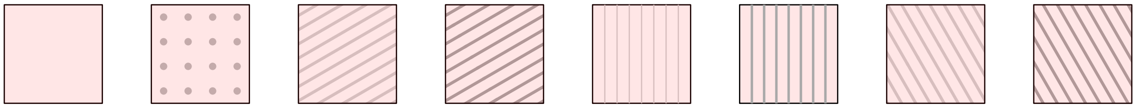


Figure 3.6: Illustration for the texture mapping for increasing data values (from left to right).

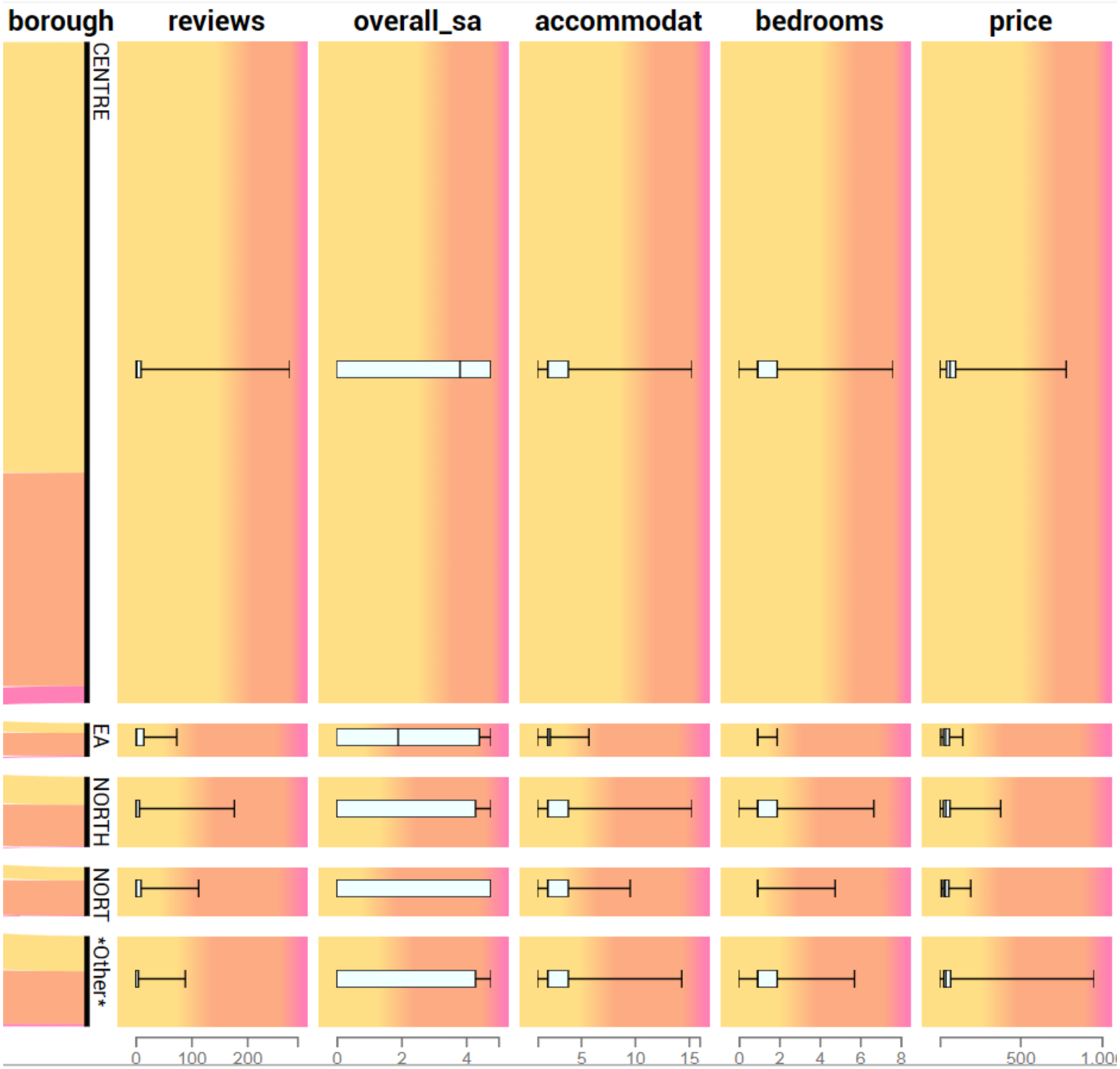


Figure 3.7: Statistics columns with box plots for each cell.

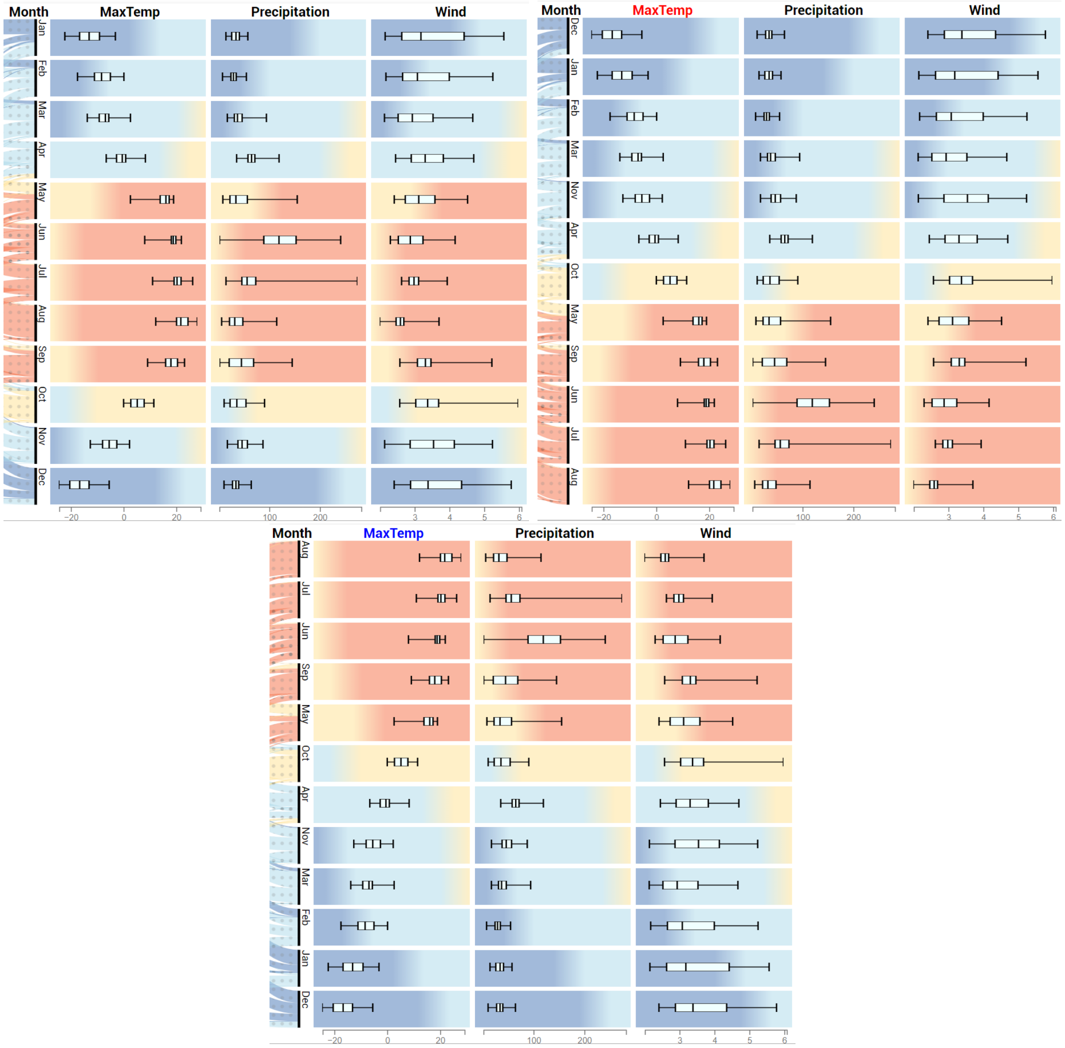


Figure 3.8: Box plot based statistics columns in unchanged monthly order (upper-left), sorted by MaxTemp in ascending order (upper-right), and sorted by MaxTemp in descending order (bottom).

the attribute of the selected column.

When trying to show the color gradient background for statistics columns, we chose to use a special gradient. For example, let us assume we have four colours picked from the hue wheel with equal distances (0°, 90°, 180°, 270°) that share 10%, 20%, 30%, and 40% of the total space respectively. If we show it directly (Figure 3.9(a)), it clearly gives information that is supposed to be shown, but it looks too sharp and could make the boundaries of the foreground plots indistinguishable. To solve such a problem, we applied colour gradients between each color intersection (Figure 3.9(d)), which takes 20% space from each side. The reason why we do not use a linear gradient for all colours is because mixing colours too much can make the ratios of each colour unclear. For example, we can make a colour gradient having the first colour located at 0%, the second colour located at 10% + (20%/2), the third colour located at 10% + 20% + (30%/2), and the last colour located at 100%, which creates a gradient shown as Figure 3.9(b). It is easy to notice that the

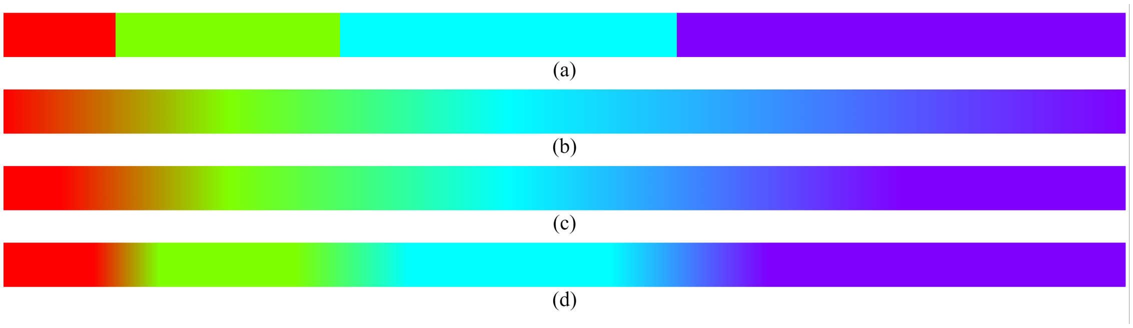


Figure 3.9: Color gradients

ratios of each colour are almost unrecognizable especially for the first and the last colour. Figure 3.9(c) demonstrates a possible improvement of Figure 3.9(b). It locates the first and the last colour to the middle of their corresponding spaces instead of the very beginning and the very end, i.e. 10%/2 instead of 0% and 100% − (40%/2) instead of 100%. It makes the first and last colour more recognizable but we still cannot tell the relatively accurate ratio that each colour takes and what the original colours are, especially when we are not sure how many colours are there and what these colours are. However, Figure 3.9(d) maintains most parts of the original colours (at least 60% of the original colours are shown without blending) and we can see a relatively accurate ratio of each colour while not making the background color intersections look too sharp.

3.3 Map View

The third component of Set-Stat-Map is a map view, which usually provides more information from another visualization perspective and gives users abilities to filter the dataset. This view is an excellent booster for visualizing datasets that contain spatial information. In this view, it could show borders, a heatmap, a few pins, etc.

For example, in a weather dataset, the data is geolocated (i.e., gridded with latitude and longitude information) and displayed using colour gradient heatmap above a Google Map layer. The heatmap relates to a specific category segment from an axis of the Parallel Sets view. (It makes the most sense when relating to segments of the last axis, which equivalently stands for the corresponding row.) Once the user selects a specific segment of a Parallel Sets axis, the heatmap cells containing those data records are displayed. Figure 3.10 is an example of a map view that displays a textured heatmap based on the Parallel Sets view's selection, which uses the colours from the first axis of the Parallel Sets and the texture patterns from the second axis of the Parallel Sets.

While putting density plots or heatmaps on maps is self-explanatory for geolocated datasets, however, the encoding used in our heatmap requires further explanation. The heatmap provides users with an idea of where the data points are located. The rationale behind using the colours and textures is to provide a sense of connection with the other components, as well as to provide users with a distribution of the colors and

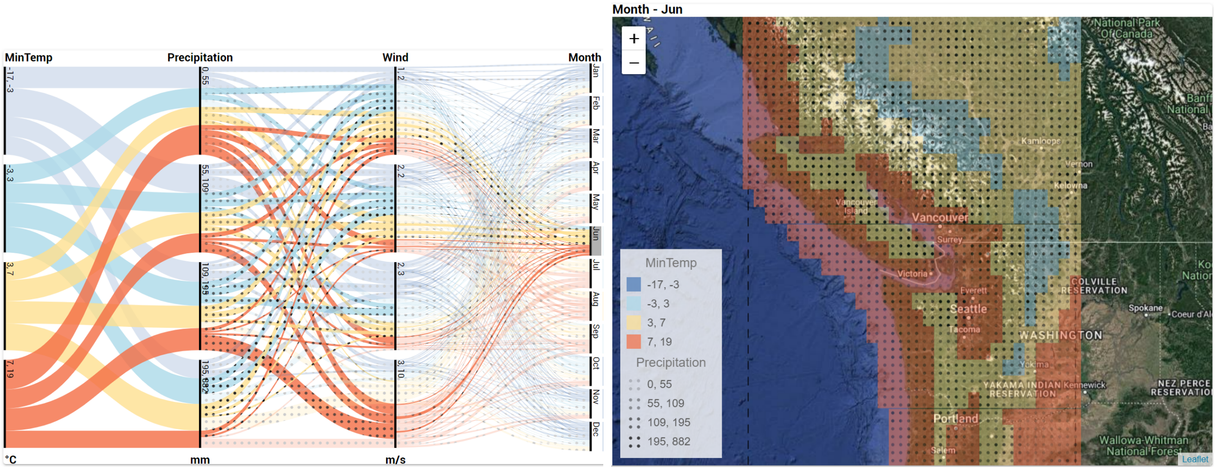


Figure 3.10: Map view (right) shows a heatmap over a Google Map layer based on the Parallel Sets view (left) selection (Month of June), where the colours and textures in the map view correspond to the highlighted ribbons of Parallel Sets view.

textures. Such a distribution over a spatial domain allows users to examine how various pairwise combinations of categories from the first and second axis relate spatially.

In addition to providing extra spacial information of the dataset, it can also act as an area filter. Figure 3.11 (top) shows the visualization without filtering the area (the black dashed lines stand for the bounding box of the dataset); Figure 3.11 (bottom) shows how the visualization is slightly changed when the area filter is applied on the map view (the grey rectangle).

3.4 Control Panel

For a specific Set-Stat-Map application, we usually provide a control panel to give users the ability to configure the visualization in detail. For different kinds of datasets, we would need to have different controls to make Set-Stat-Map to best fit our needs. Figure 3.12 (left) shows the control panel for the weather dataset. It gives users the ability to configure ribbons' opacity, texture pattern types, how the numerical values should be grouped, which variables to be visualized, the order of visualized variables, and if an additional visualization for comparison should be shown. Figure 3.12 (right) demonstrate the control panel for a StackOverflow dataset.

3.5 Coordination

We coordinate the views by synchronizing the updates on different views. The standard interactions on the Parallel Sets [40] have also been implemented in our design. Since the colors and textures on the ribbons are based on the first and second axis, one can reorder the axis and choose the first or second axis based on their need. Such an interaction simultaneously updates the colors of the statistics columns and heatmap.

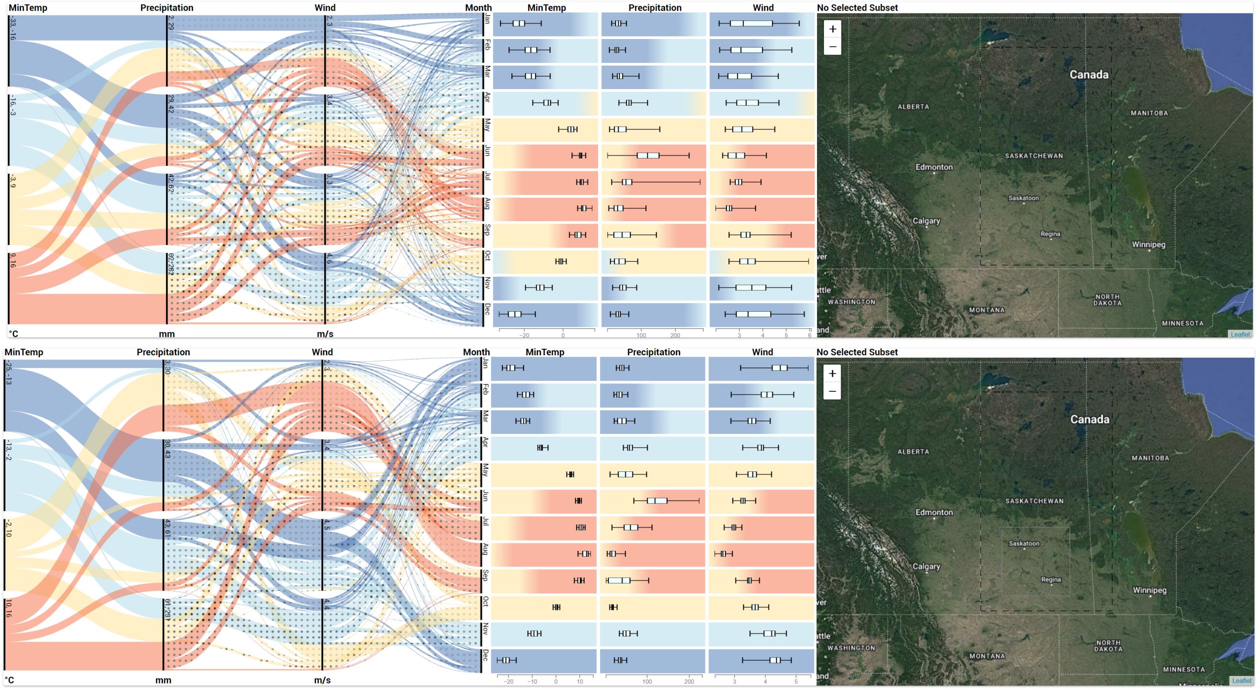
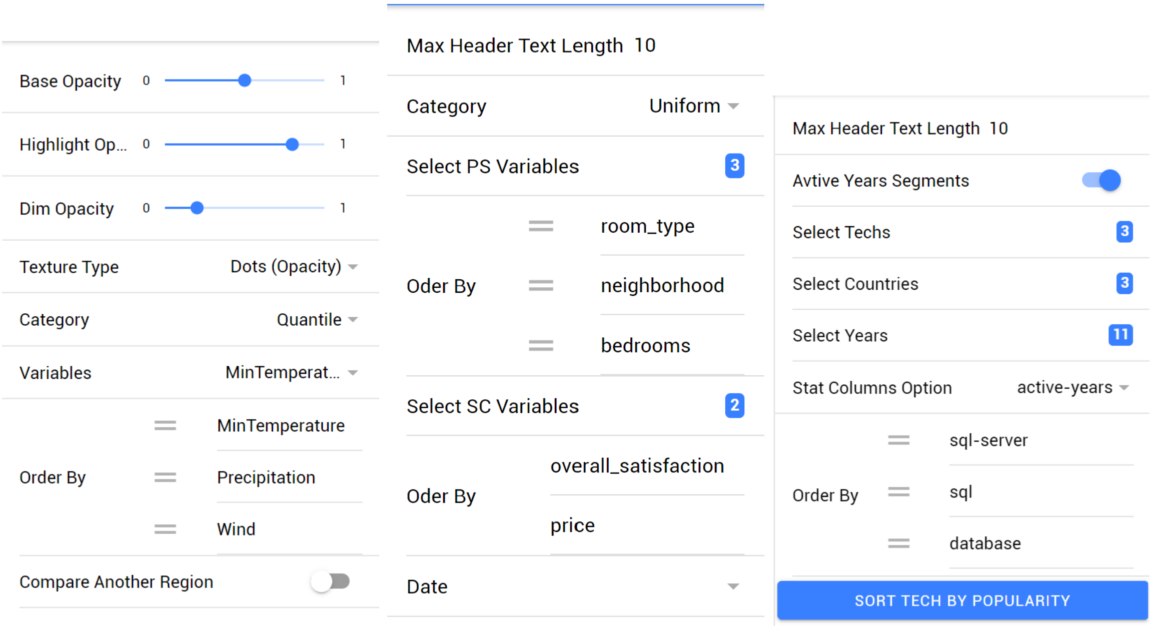


Figure 3.11: No area filter is applied (top) and an area filter is applied (bottom)



Base Opacity 0 1

Highlight Op... 0 1

Dim Opacity 0 1

Texture Type Dots (Opacity)

Category Quantile

Variables MinTemperat...

Order By
MinTemperature
Precipitation
Wind

Compare Another Region

Max Header Text Length 10

Category Uniform

Select PS Variables 3

Oder By
room_type
neighborhood
bedrooms

Select SC Variables 2

Oder By
overall_satisfaction
price

Date

Max Header Text Length 10

Avtive Years Segments

Select Techs 3

Select Countries 3

Select Years 11

Stat Columns Option active-years

Order By
sql-server
sql
database

SORT TECH BY POPULARITY

Figure 3.12: Control panel for the weather dataset (left), the Airbnb dataset (middle), and for the StackOverflow dataset (right)

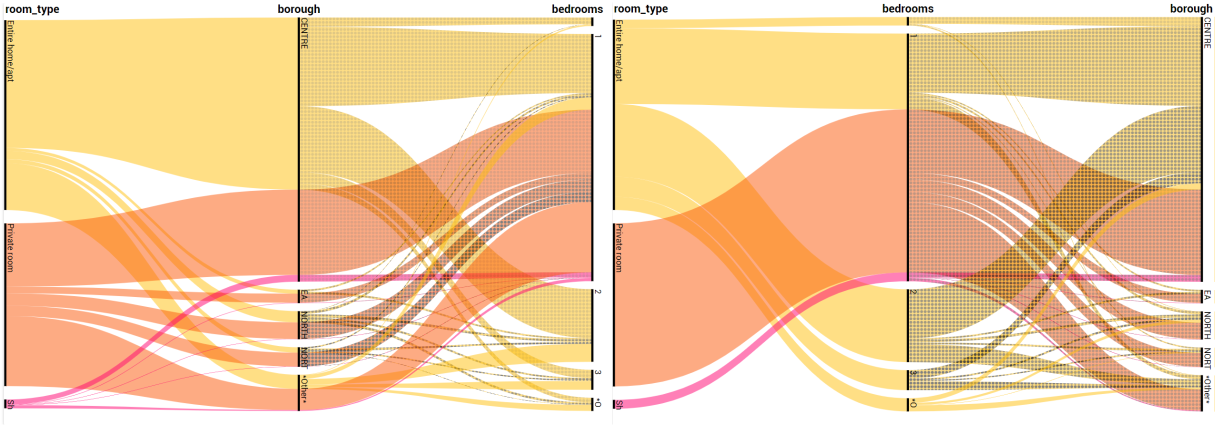


Figure 3.13: Example of Parallel Sets view using different axis ordering (note that the statistic columns view will also change according to the rightmost axis).

Similarly, changing the last axis of the Parallel Sets view changes the number of rows in the statistics column based on the categories on the new last axis.

For example, changing the last axis of Figure 3.1 from monthly to seasonal (spring, summer, fall, and winter) would create 4 rows in the statistics columns view instead of the 12 rows that it currently shows. Removal of the month axis and using Wind as the last axis would also create 4 rows corresponding to the 4 intervals at the Wind axis. This happens also for the reordering of axes, see Figure 3.13.

A statistics column can be sorted by clicking on the column header. For different datasets or scenarios, implementing the sorting method based on different statistical attributes (i.e., mean, medium, min, max, etc.) might be useful. Sorting a column will reorder both the rows and the intervals of the last axis of the Parallel Sets view. For example, in Figure 3.2, the MaxTemp column is sorted and the months are reordered based on this sorting (instead of being chronologically ordered from Jan. to Dec.). The users can filter the data records by drawing rectangles on the map view, which updates the ribbons in the Parallel Sets view and the plots on the statistics columns (see Figure 3.11).

3.6 Design Rationale

The Set-Stat-Map design follows Baldonado's [19] guidelines for designing multiple coordinated views. The three fundamentally different types of information (categorical, numeric, and geolocation) justify leveraging the three views (Parallel Sets, statistics columns, and map). This follows the rules of *decomposition, complementarity* and *parsimony*, which suggests partitioning complex data into multiple views to ease user comprehension, to provide complementary information to best leverage each view, and to use the views minimally (avoiding adding a view unless there is a significant benefit that justifies the cost).

The Set-Stat-Map design also follows the rule of *resource optimization*. We organize the views side-by-side in a linear fashion, which creates a potential to provide (static or user interaction based) text summary beneath the display. In addition, the user can compare two Set-Stat-Map visualizations by placing them one on top of the other (Figure 3.11), which is useful to compare two map selections or two different datasets. Furthermore, we limit the number of categories visible simultaneously to maintain the system's interactivity. The flow of colour and texture from ribbons to the statistics columns view's rows and finally, to the map make relationships among multiple views apparent to the user, which is aligned with the rule of *self-evidence*. The rule of *consistency* can be observed from the synchronized view updates upon user interactions (axis reordering, row sorting and map selection).

A novel interaction that Set-Stat-Map introduces is interval selection. By selecting a vertical interval of an axis in the Parallel Sets view, the user can render the information (data records) of all the ribbons passing through that interval on the map. For the weather dataset, selecting such an interval (e.g., May) on the last axis (Month) would render all the tiles corresponding to this month on the map. We can thus see the spatial distribution of the first and second axis through color and texture distribution, respectively. This is illustrated in Figure 3.14(top), where by selecting the interval May at the Month axis, we can observe higher MaxTemp and precipitation at the south-west region and the opposite trend at the north-east region. In fact, this selection operation generalizes to the intervals of all the axes of the Parallel Sets, e.g., selection of the second interval of the Relative Humidity axis (Figure 3.14(bottom)) blends all the tiles corresponding to that interval over all months to depict a yearly trend. The map view of Figure 3.14(bottom) illustrates the yearly distribution of the first axis (MaxTemp) through the color distribution on the map, where we can immediately see a high variation of temperature within the selected humidity range at the north-east map region. However, using such an interval selection in the intermediate axis would require using simple textures such as dots to avoid visual clutter due to texture overlap.

3.7 Summary

We design Set-Stat-Map as a three-view interactive visualization. The first view is the Parallel Sets view with vertical axes, which gives it the ability to connect to the second view. We transform numerical attributes into categorical attributes by grouping ranges of numerical attributes into one category to better suit the Parallel Sets technique. We also enhance the Parallel Sets view by adding textures for the second axis to better help pairwise comparisons, merging less important categories into one to reduce the cognitive loads, and limiting the max size of the merged category to help users to better focus on more important categories. The second view is the statistics columns view that gives additional statistics information for specific categories by box plot or bar plot on the background of color gradients relating to the Parallel Sets view. The third view is the map view, which is normally a geographical map with overlays of Parallel Sets view's colours and textures to provide spatial information about the dataset. By interacting with each view, other views can be updated based on users' operations. Thus, Set-Stat-Map is able to give information about both the big picture and details with spatial representations for a mixed-type multi-dimensional dataset.

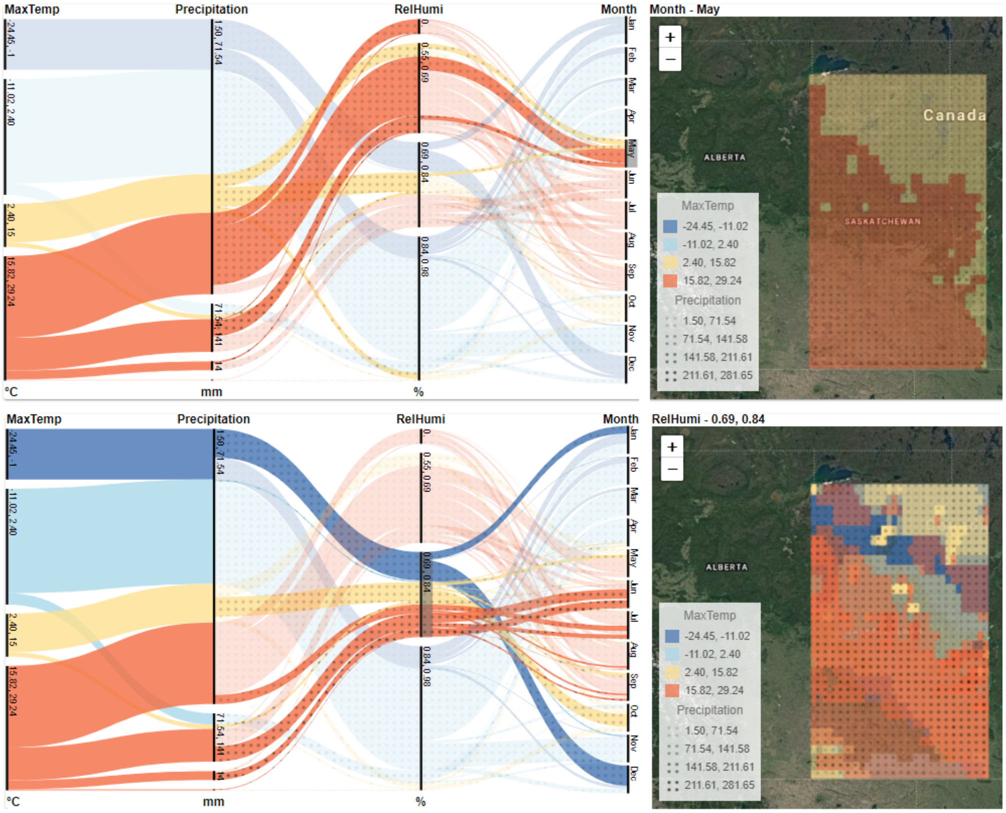


Figure 3.14: Interval selection in Parallel Sets: (top) selection of a temporal range, and (bottom) selection of a variable value range. The statistics columns are omitted for space constraints.

4 Implementation

Our proposed solution includes at least three linked interactive views – an augmented Parallel Sets view, a statistics columns view, and a map view – to better visualize the data. The augmented Parallel Sets view provides the general idea of the datasets such as the structure, the distribution for each attribute, etc. The statistics columns add additional information on the numerical attributes for the last axis subsets. The map view can be designed differently based on different datasets, which can be a heatmap, geographic map, etc.

4.1 Used Technologies

Web Component: We implemented Set-Stat-Map using multiple web components. A web component bundles some HTML/CSS/JavaScript together into a JavaScript module that most modern browsers support natively. Each web component, which contains its own logic and view, can expose attributes, properties, and events to achieve the need for configuration and customization. As long as a developer is developing a modern web app, no matter whether the project used any front-end framework (such as React, Angular, Vue, etc.) or even just used vanilla HTML/CSS/JavaScript, the developer can easily integrate web components into the project. We chose to use web components for maximizing compatibility and reusability.

StencilJS: To make the development more efficient and maintainable, we used a web component compiler called StencilJS [11]. It provides some features that usually come with some front-end frameworks, such as using templates, virtual DOM, async rendering, reactive data-binding, TypeScript, JSX, and static site generation. However, instead of running a framework instance at the runtime, it compiles all codes into the standards-based web components without unnecessary framework dependencies.

D3.js: For creating visual elements, we used D3.js [3], a popular JavaScript library to build interactive visualizations, to better operate with SVGs and add interactive events for our visualization. D3.js also provides some data processing functions to let us efficiently process and reformat datasets.

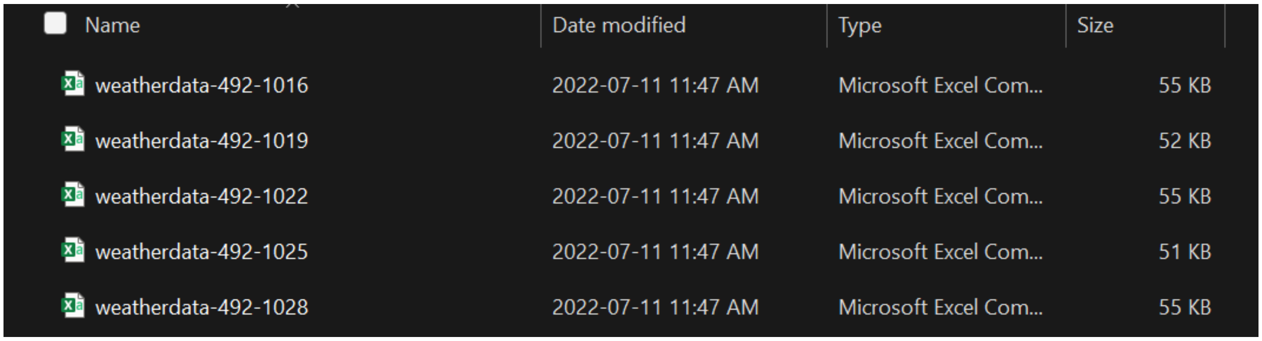
SQLite: Using databases is a great way to quickly and easily store and access large structured datasets, yet traditional databases require a server to keep running. Using serialized files is another great way to store and access data, but it usually requires developers to design all the details themselves and cannot take advantage of SQL to achieve better accessibility and performance. SQLite [8] is a file-based database, it simply uses a data file and also provides the basic features of a SQL database. By using SQLite, we can store each dataset into a single file, which can be opened by applications and efficiently queried with SQL.

SQL.js: We used SQL.js [9] for creating and querying SQLite database files directly on the browser. SQL.js ports SQLite into web environment by compiling it as WebAssembly. WebAssembly is compatible with modern browsers, providing better performance than JavaScript, and can be compiled from non-web languages (such as C, C++, and Rust).

4.2 Datasets

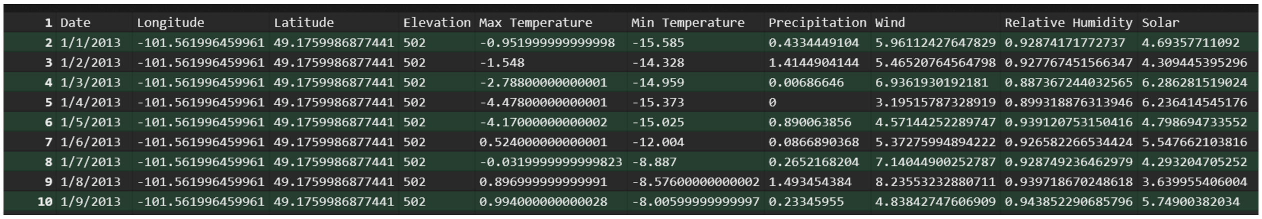
For each dataset, We first obtain the raw data. After that, we write a script (for each type of data) to process and re-format the data. Then, the data is maintained using a single SQLite database file, containing a single table with columns for all necessary attributes.

Meteorological Dataset: The raw data is obtained from *Global Weather Data for SWAT* [4], which is a website providing CFSR (Climate Forecast System Reanalysis) data from 1979 to 2014. By using their tool, we choose a rectangular area and a time interval that we are interested in. Then we get a set of CSV files from the website. For each selection, the website gives us a ZIP file that contains a set of CSV files (Figure 4.1). Each CSV file is named with its coordinate, stands for a location point, and contains its date, coordinate (latitude and longitude), elevation(m), max/min temperatures($^\circ C$), precipitation(mm), wind level(m/s), relative humidity($fraction$), and solar level(MJ/m^2) (Figure 4.1). We use a script to merge all files together and calculate the average or sum (depending on the attribute type) of all days for each month, which transformed the Date attribute into a Month attribute, to produce a single-table SQLite file (Figure 4.3). The longitudes and latitudes of each data point are approximately 36 km away from the other data point (data are at 36 km spatial resolution). Most attributes are numeric values for each date and geolocation, and hence ideal to test the functionality of Set-Stat-Map on a predominately numeric dataset with spatial attributes.



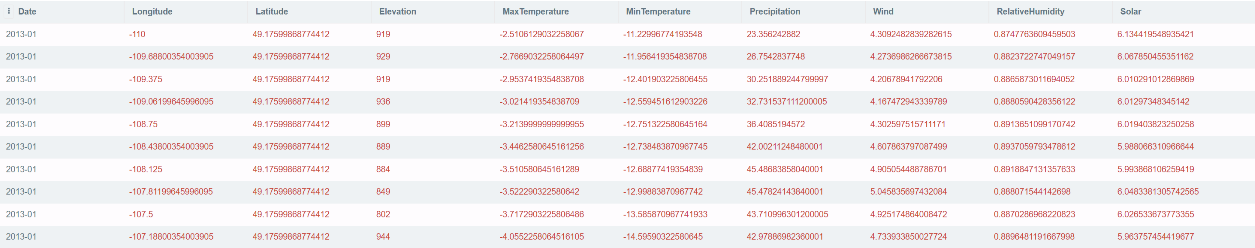
Name	Date modified	Type	Size
weatherdata-492-1016	2022-07-11 11:47 AM	Microsoft Excel Com...	55 KB
weatherdata-492-1019	2022-07-11 11:47 AM	Microsoft Excel Com...	52 KB
weatherdata-492-1022	2022-07-11 11:47 AM	Microsoft Excel Com...	55 KB
weatherdata-492-1025	2022-07-11 11:47 AM	Microsoft Excel Com...	51 KB
weatherdata-492-1028	2022-07-11 11:47 AM	Microsoft Excel Com...	55 KB

Figure 4.1: Sample of CSV files of the raw weather dataset.



1	Date	Longitude	Latitude	Elevation	Max Temperature	Min Temperature	Precipitation	Wind	Relative Humidity	Solar
2	1/1/2013	-101.561996459961	49.1759986877441	502	-0.951999999999998	-15.585	0.4334449104	5.96112427647829	0.92874171772737	4.69357711092
3	1/2/2013	-101.561996459961	49.1759986877441	502	-1.548	-14.328	1.4144904144	5.46520764564798	0.927767451566347	4.309445395296
4	1/3/2013	-101.561996459961	49.1759986877441	502	-2.78800000000001	-14.959	0.00686646	6.9361930192181	0.887367244032565	6.286281519024
5	1/4/2013	-101.561996459961	49.1759986877441	502	-4.47800000000001	-15.373	0	3.19515787328919	0.899318876313946	6.236414545176
6	1/5/2013	-101.561996459961	49.1759986877441	502	-4.17000000000002	-15.025	0.890063856	4.57144252289747	0.939120753150416	4.798694733552
7	1/6/2013	-101.561996459961	49.1759986877441	502	0.524000000000001	-12.004	0.0866890368	5.37275994894222	0.926582266534424	5.547662103816
8	1/7/2013	-101.561996459961	49.1759986877441	502	-0.0319999999999823	-8.887	0.2652168204	7.14044900252787	0.928749236462979	4.293204705252
9	1/8/2013	-101.561996459961	49.1759986877441	502	0.896999999999991	-8.57600000000002	1.493454384	8.23553232880711	0.939718670248618	3.639955406004
10	1/9/2013	-101.561996459961	49.1759986877441	502	0.994000000000028	-8.00599999999997	0.23345955	4.83842747606909	0.943852290685796	5.74900382034

Figure 4.2: Sample of a CSV file of the raw weather dataset.



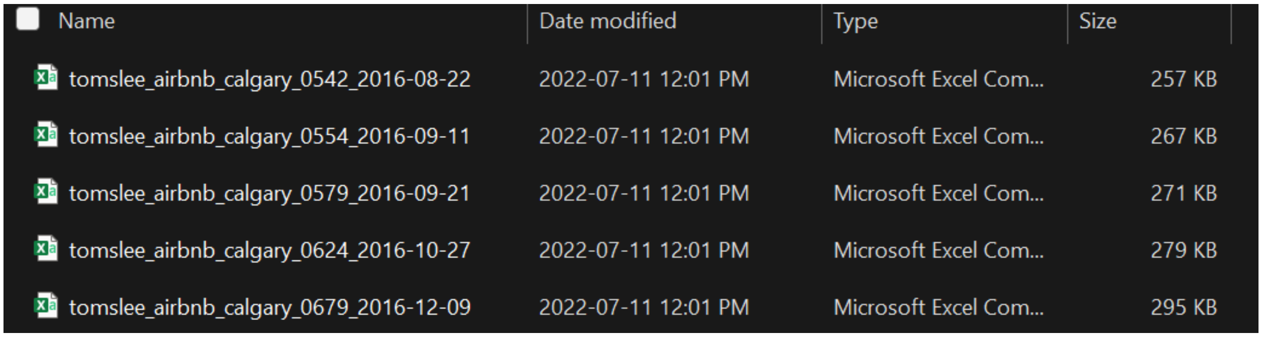
Date	Longitude	Latitude	Elevation	MaxTemperature	MinTemperature	Precipitation	Wind	RelativeHumidity	Solar
2013-01	-110	49.17599868774412	919	-2.5106129032258067	-11.22996774193548	23.356242882	4.3092482839282615	0.8747763609459503	6.134419548935421
2013-01	-109.68800354003905	49.17599868774412	929	-2.7669032258064497	-11.956419354838708	26.7542837748	4.2736986266673815	0.8823722747049157	6.067850455351162
2013-01	-109.375	49.17599868774412	919	-2.9537419354838708	-12.401903225806455	30.251889244799997	4.20678941792206	0.8865873011694052	6.010291012869869
2013-01	-109.06199645996095	49.17599868774412	936	-3.021419354838709	-12.559451612903226	32.731537111200005	4.167472943339789	0.8880590428356122	6.01297348345142
2013-01	-108.75	49.17599868774412	899	-3.2139999999999955	-12.751322580645164	36.4085194572	4.302597515711171	0.8913651099170742	6.019403823250258
2013-01	-108.43800354003905	49.17599868774412	889	-3.4462580645161256	-12.738483870967745	42.00211248480001	4.607863797087499	0.8937059793478612	5.988066310966644
2013-01	-108.125	49.17599868774412	884	-3.510580645161289	-12.68877419354839	45.48683858040001	4.905054488786701	0.8918847131357633	5.993868106259419
2013-01	-107.81199645996095	49.17599868774412	849	-3.522290322580642	-12.99883870967742	45.47824143840001	5.045835697432084	0.888071544142698	6.0483381305742565
2013-01	-107.5	49.17599868774412	802	-3.7172903225806486	-13.585870967741933	43.710996301200005	4.925174864008472	0.8870286968220823	6.026533673773355
2013-01	-107.18800354003905	49.17599868774412	944	-4.0552258064516105	-14.59590322580645	42.97886982360001	4.733933850027724	0.8896481191667998	5.963757454419677

Figure 4.3: Sample of the process SQLite file of weather dataset.

Airbnb Dataset: Airbnb is an online vacation rental service. We use a dataset for listings over Calgary, which contained 2521 listings from Dec 12, 2019. We get a set of CSV files (Figure 4.4) from *Tom Slee* [12, 1], each of which contains data of a specific date. For each date, it gives the following information (Figure 4.5). See [1] for the details.

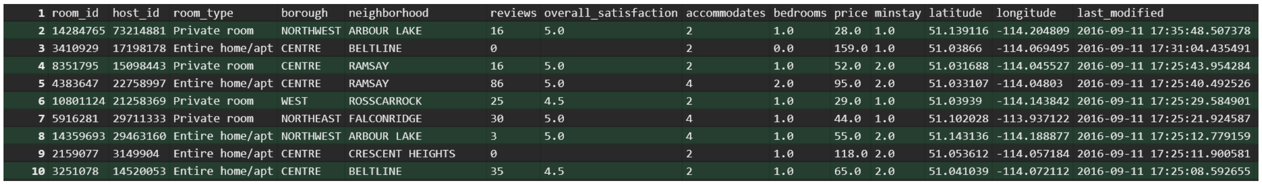
- room_id: This is the unique identifier for a listing.
- host_id: This is the unique identifier for a host.
- room_type: Could be "Entire home/apt", "Private room", or "Shared room"
- borough: A sub-region of the city.
- neighborhood: A sub-region of the city.
- reviews: The number of reviews.
- overall_satisfaction: The average rating (out of five) of reviews.
- accommodates: The capacity of guests.
- bedrooms: The number of bedrooms.
- price: The price (in $US) for a night.
- minstay: The minimum days to stay.
- latitude and longitude: Coordinate of the room.
- last_modified: The time that the data is fetched from the Airbnb site.

We then merge all CSV files together and store them into an SQLite database file (Figure 4.6).



Name	Date modified	Type	Size
tomslee_airbnb_calgary_0542_2016-08-22	2022-07-11 12:01 PM	Microsoft Excel Com...	257 KB
tomslee_airbnb_calgary_0554_2016-09-11	2022-07-11 12:01 PM	Microsoft Excel Com...	267 KB
tomslee_airbnb_calgary_0579_2016-09-21	2022-07-11 12:01 PM	Microsoft Excel Com...	271 KB
tomslee_airbnb_calgary_0624_2016-10-27	2022-07-11 12:01 PM	Microsoft Excel Com...	279 KB
tomslee_airbnb_calgary_0679_2016-12-09	2022-07-11 12:01 PM	Microsoft Excel Com...	295 KB

Figure 4.4: Sample of CSV files of the raw Airbnb dataset.



room_id	host_id	room_type	borough	neighborhood	reviews	overall_satisfaction	accommodates	bedrooms	price	minstay	latitude	longitude	last_modified
14284765	73214881	Private room	NORTHWEST	ARBOUR LAKE	16	5.0	2	1.0	28.0	1.0	51.139116	-114.204809	2016-09-11 17:35:48.507378
3410929	17198178	Entire home/apt	CENTRE	BELTLINE	0		2	0.0	159.0	1.0	51.03866	-114.069495	2016-09-11 17:31:04.435491
8351795	15098443	Private room	CENTRE	RAMSAY	16	5.0	2	1.0	52.0	2.0	51.031688	-114.045527	2016-09-11 17:25:43.954284
4383647	22758997	Entire home/apt	CENTRE	RAMSAY	86	5.0	4	2.0	95.0	2.0	51.033107	-114.04803	2016-09-11 17:25:40.492526
10801124	21258369	Private room	WEST	ROSSCARROCK	25	4.5	2	1.0	29.0	1.0	51.03939	-114.143842	2016-09-11 17:25:29.584901
5916281	29711333	Private room	NORTHEAST	FALCONRIDGE	30	5.0	4	1.0	44.0	1.0	51.102028	-113.937122	2016-09-11 17:25:21.924587
14359693	29463160	Entire home/apt	NORTHWEST	ARBOUR LAKE	3	5.0	4	1.0	55.0	2.0	51.143136	-114.188877	2016-09-11 17:25:12.779159
2159077	3149904	Entire home/apt	CENTRE	CRESCENT HEIGHTS	0		2	1.0	118.0	2.0	51.053612	-114.057184	2016-09-11 17:25:11.900581
3251078	14520053	Entire home/apt	CENTRE	BELTLINE	35	4.5	2	1.0	65.0	2.0	51.041039	-114.072112	2016-09-11 17:25:08.592655

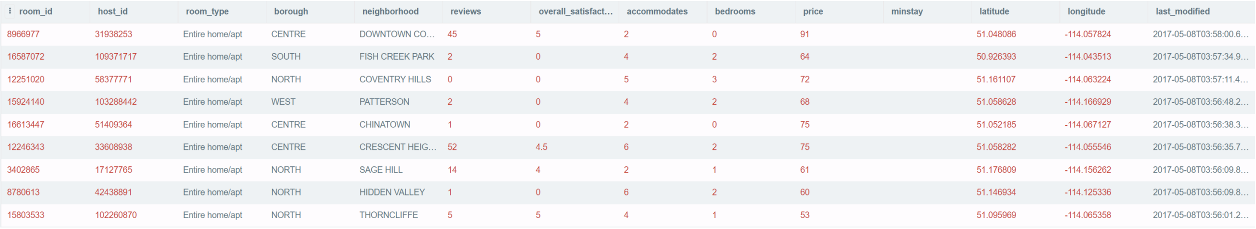
Figure 4.5: Sample of a CSV file of the raw Airbnb dataset.

StackOverflow Dataset: The raw data is obtained from StackOverflow from 2008 to 2018, which contains all users' posts of questions and answers [45]. We get a single CSV file (Figure 4.7), which has columns of id (a unique identifier), profile creation date, developer name, location of the developer, and the technologies used for each year (from 2008 to 2018, each year as a separate column). Then we process and re-structure it to generate an SQLite dataset file (Figure 4.8), which only has four columns (UserId, Tech, Year, and Location) for easy understanding and querying.

4.3 Development Iterations (Meteorological Dataset)

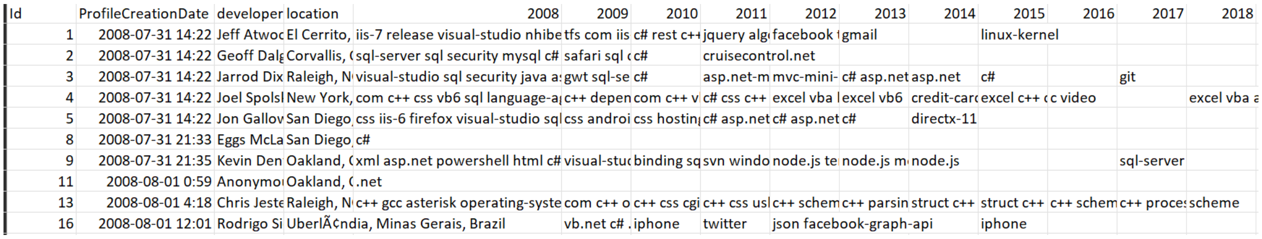
To ensure Set-Stat-Map can be helpful in real-life cases, we initiated a collaboration with two environmental scientists: one within the School of Environment and Sustainability, and another scientist from the Global Institute for Water Security. Due to their field of research, the cooperation focuses mainly on weather dataset visualization.

We first demonstrate our system to them, and we observed the need for several meetings so that they become comfortable with using the system. We set weekly one-hour meetings over three months so that they can suggest modifications to SET-STAT-MAP. A major challenge that we noticed is that the experts took about three to four meetings before they could comprehend the system well enough to reason about it spontaneously. Initially, they had several questions on why the number of crossings increases as we move from the first to the later axes, and whether these crossings mean anything, and how they relate to the statistics column, what the colors in the statistics columns mean and so on. The Parallel Set design was not something that they were familiar with, and it was not straightforward for them to understand the connections among different views. It also took some time for the experts to recall the system interactions and effectively use



room_id	host_id	room_type	borough	neighborhood	reviews	overall_satisfact…	accommodates	bedrooms	price	minstay	latitude	longitude	last_modified
8966977	31938253	Entire home/apt	CENTRE	DOWNTOWN CO…	45	5	2	0	91		51.048086	-114.057824	2017-05-08T03:58:00.6…
16587072	109371717	Entire home/apt	SOUTH	FISH CREEK PARK	2	0	4	2	64		50.926393	-114.043513	2017-05-08T03:57:34.9…
12251020	58377771	Entire home/apt	NORTH	COVENTRY HILLS	0	0	5	3	72		51.161107	-114.063224	2017-05-08T03:57:11.4…
15924140	103288442	Entire home/apt	WEST	PATTERSON	2	0	4	2	68		51.058628	-114.166929	2017-05-08T03:56:48.2…
16613447	51409364	Entire home/apt	CENTRE	CHINATOWN	1	0	2	0	75		51.052185	-114.067127	2017-05-08T03:56:38.3…
12246343	33608938	Entire home/apt	CENTRE	CRESCENT HEIG…	52	4.5	6	2	75		51.058282	-114.055546	2017-05-08T03:56:35.7…
3402865	17127765	Entire home/apt	NORTH	SAGE HILL	14	4	2	1	61		51.176809	-114.156262	2017-05-08T03:56:09.8…
8780613	42438891	Entire home/apt	NORTH	HIDDEN VALLEY	1	0	6	2	60		51.146934	-114.125336	2017-05-08T03:56:09.8…
15803533	102260870	Entire home/apt	NORTH	THORNCLIFFE	5	5	4	1	53		51.095969	-114.065358	2017-05-08T03:56:01.2…

Figure 4.6: Sample of the process SQLite file of Airbnb dataset.



Id	ProfileCreationDate	developer	location	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018
1	2008-07-31 14:22	Jeff Atwoc	El Cerrito,	iis-7 release visual-studio nhibe	tfs com iis	c# rest c++	jquery alg	facebook t	gmail		linux-kernel			
2	2008-07-31 14:22	Geoff Dalg	Corvallis, (	sql-server sql security mysql c#	safari sql c	c#	cruisecontrol.net							
3	2008-07-31 14:22	Jarrod Dix	Raleigh, N	visual-studio sql security java a:	gwt sql-se	c#	asp.net-m	mvc-mini-	c# asp.net	asp.net	c#		git	
4	2008-07-31 14:22	Joel Spolsl	New York,	com c++ css vb6 sql language-a	c++ depen	com c++ v	c# css c++	excel vba l	excel vb6	credit-carc	excel c++ (	c video		excel vba a
5	2008-07-31 14:22	Jon Gallov	San Diego,	css iis-6 firefox visual-studio sql	css androi	css hostin	c# asp.net	c# asp.net	c#	directx-11				
8	2008-07-31 21:33	Eggs McLa	San Diego,	c#										
9	2008-07-31 21:35	Kevin Den	Oakland, (	xml asp.net powershell html c#	visual-stuc	binding sq	svn windo	node.js te	node.js m	node.js			sql-server	
11	2008-08-01 0:59	Anonymou	Oakland, (	.net										
13	2008-08-01 4:18	Chris Jeste	Raleigh, N	c++ gcc asterisk operating-syste	com c++ o	c++ css cgi	c++ css usl	c++ schem	c++ parsin	struct c++	struct c++	c++ schem	c++ proce:	scheme
16	2008-08-01 12:01	Rodrigo Si	UberlÃ¢ndia, Minas Gerais, Brazil		vb.net c# .	iphone	twitter	json facebook-graph-api			iphone			

Figure 4.7: Sample of a CSV file of the raw StackOverflow dataset.

Set-Stat-Map.

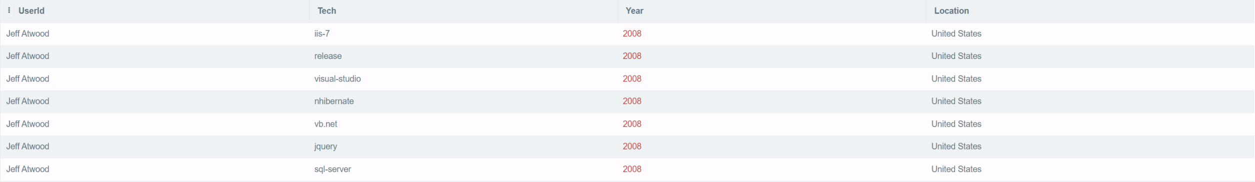
After a few meetings, the experts were able to explore our system by trying to accomplish some research tasks and to give us feedback according to their aspects. We then iterate our system to a new version based on their feedback and let them explore it again. Thanks to the experts' feedback, we improved our system in several ways that we could not have thought about.

4.3.1 Adding Extra Information for the Parallel Sets Axes

We realized that the experts would care about the unit of each attribute in the weather dataset as there could be multiple possible units for an attribute we provide. For example, the temperature could be in either Celsius or Fahrenheit. Thus, we need to provide a clear hint about the unit of each attribute for them. We first thought about adding a tooltip, which shows up when the user hovers on an axis' header, to give information about the unit, but then we realized that is not obvious and intuitive enough, so we finally chose to make use of the bottom space of the axes to show the corresponding units.

4.3.2 Adding Numeric Axes for Statistics Columns

The statistics columns view is designed for users to obtain statistical information about the sets of the last axis of the Parallel Sets. One thing the experts complain about is the fact that even though the statistics view is only able to get some basic distribution information, it is too hard to figure out the value ranges. To find the value range of a statistics column, they have to check back the minimum and maximum values from the corresponding Parallel Sets axis, which is non-intuitive and time-consuming. So they suggested adding numeric axes on the bottom of the statistics column view, which can also fill the space that is horizontally aligned to the units of attributes described in the previous section.



⁝ UserId	Tech	Year	Location
Jeff Atwood	iis-7	2008	United States
Jeff Atwood	release	2008	United States
Jeff Atwood	visual-studio	2008	United States
Jeff Atwood	nhibernate	2008	United States
Jeff Atwood	vb.net	2008	United States
Jeff Atwood	jquery	2008	United States
Jeff Atwood	sql-server	2008	United States

Figure 4.8: Sample of the process SQLite file of StackOverflow dataset.

4.3.3 Using Different Colour Schemes for Attributes

We initially only provided one single colour scheme for all attributes, and experts argued that it could be non-intuitive and increase the cognitive load for users. For example, it looks good when we use blue-to-red schema for low-to-high temperature, but it would not work well for precipitation as blue can lead people to think about a higher level of precipitation. So they suggested we use different colour schemes for attributes and provided us with some colour schemes that they prefer. Figure 4.9 shows three colour schemes for different attributes.

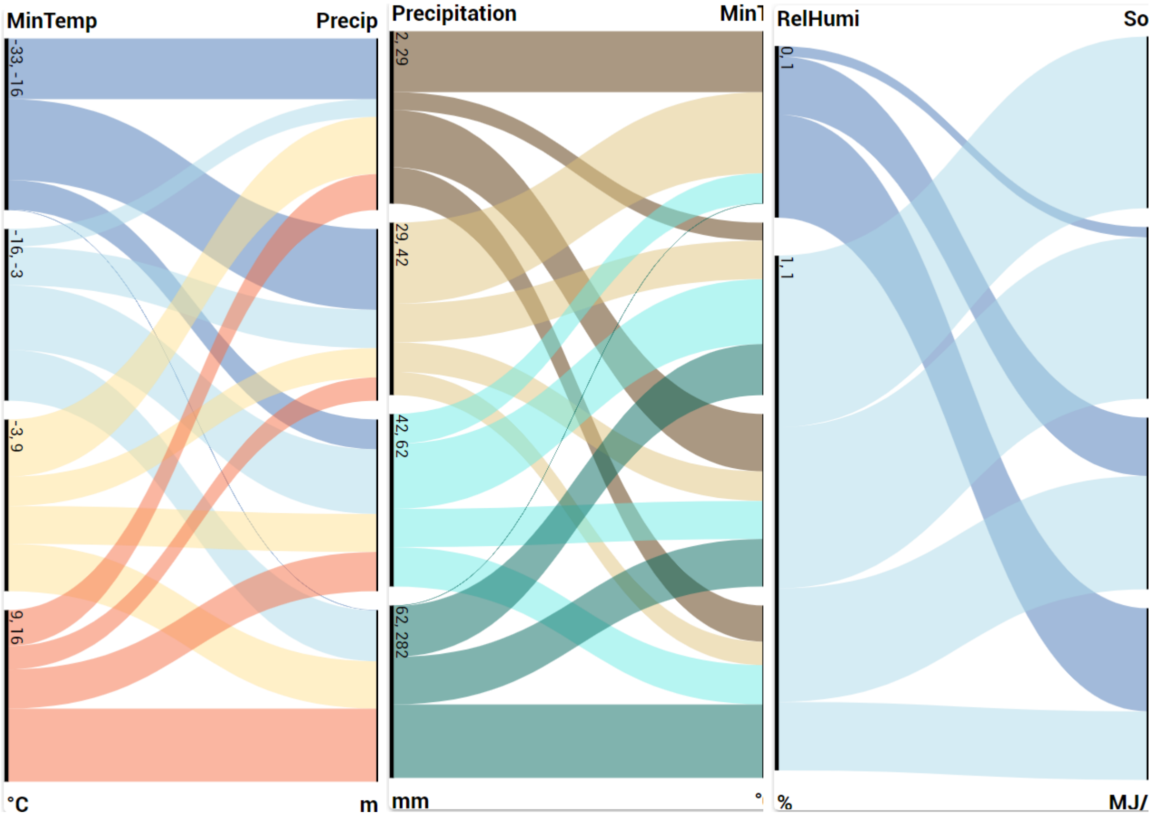


Figure 4.9: Parallel Sets view uses different colour Schemes for temperature (left), precipitation (middle), and relative humidity (right).

4.3.4 Other Improvements

The experts also let us know about some of their preferences and we used these to set our default visualization configurations. For example, we provided some texture combination options for the visualization and we picked dots with different opacity as the default configuration for weather dataset visualization. We also removed some combinations that cause confusion such as arbitrary textures and some complex textures. Another example is that even though we provide two ways to group numerical values, the experts prefer to use quartiles. These details make the experts feel easier to understand the data (due to the smaller number of ribbons, less clutter, and faster interactions).

4.4 Development for Other Datasets

For the Airbnb dataset and StackOverflow dataset, we created a SET-STAT-MAP based on the natural properties of the dataset and discussed that with some general participants to examine its generalizability. For these datasets, we did not perform any development iterations.

4.5 Summary

We implement SET-STAT-MAP as a pure web based app and take advantage of technologies such as web components, StencilJS, D3.js, SQLite, SQL.js, etc. For our three datasets, we process their raw data and export each dataset as an SQLite database file for later use by the web app. During the development, we collaborate with experts to constantly receive their feedback. This feedback helps us to better figure out their needs and how to improve our visualization's usability. We use this feedback for our development iterations and made changes such as adding a display for each attribute unit, adding numerical axes for statistics columns, using different colour schemes for attributes, etc.

5 Evaluations

Set-Stat-Map can handle a wide range of datasets with numeric and categorical attributes, even with geolocation information. We first examine how the performances and the visual properties of the system vary with the changes in data dimensions (Section 5.1). We next describe our interaction with an environmental scientist (for Weather data), with a professional with some knowledge of Airbnb and web development (Airbnb dataset) and potential usage challenges with the StackOverflow dataset (Section 5.2).

5.1 Performance Evaluation

To evaluate the performance of Set-Stat-Map, we analyzed its performance with the meteorological dataset [4] on an Intel i5-1135G7 machine with 16GB RAM, where the visualization was loaded with MS Edge 97.0.1072.55 on a 4K monitor.

5.1.1 Experimental Setup

Since different variable selections can have different visual complexities in the final visualization, we first randomly picked three ordering of the five variables as below:

1. MaxTemp, Precipitation, Wind, RelHumi, and Solar

2. Elevation, RelHumi, Wind, Precipitation, and Solar

3. Solar, MaxTemp, MinTemp, Wind, and RelHumi

We want to evaluate how the performances vary over the choice of quartile and uniform categorization of the variables, as well as when we change the variable count from 2 to 5. Therefore, for each ordering and for each combination of variable count and categorization, we generated Set-Stat-Map visualization 5 times.

We gather our raw data in two categories: visual complexity and time complexity. To evaluate the visual complexity, between every two axes, we gather information about the number of ribbons that are displayed in the Parallel Sets and the number of ribbons that they are crossing each other. For the time complexity, we consider some potential time-consuming parts: the Parallel Sets rendering, the statistics columns rendering, data querying, and categorizing data for uniform or quartiles. We record the time of these for each run.

5.1.2 Results

After gathering the raw data, we compute some statistics and group the data by variable counts (from 2 to 5) for each measured information. As a measure of visual complexity, we use ribbon count and ribbon crossing count, and calculate summary statistics (average, minimum, and maximum) of three variable orderings. For time measurements, we first calculated the averages of 5 runs for each configuration, and then we calculated the statistics (average, minimum, and maximum) of three variable orderings. With these statistics, we plot the statistics of each measurement by variable counts:

- the number of ribbon count (Figure 5.1)
- the number of ribbon crossing count (Figure 5.2)
- the time of rendering the Parallel Sets (Figure 5.3)
- the time of rendering the statistics columns (Figure 5.4)
- the time of querying data (Figure 5.5)
- the time of categorization for uniform and quartile (Figure 5.6)

In each plot, the measurement averages are shown as bars (blue for uniform categorizing and red for quartile categorizing); the standard deviation is shown as the black lines over the corresponding bars. In addition to these, we also make a table about the time taken by the potential time-consuming parts for different variable counts as Table 5.1.

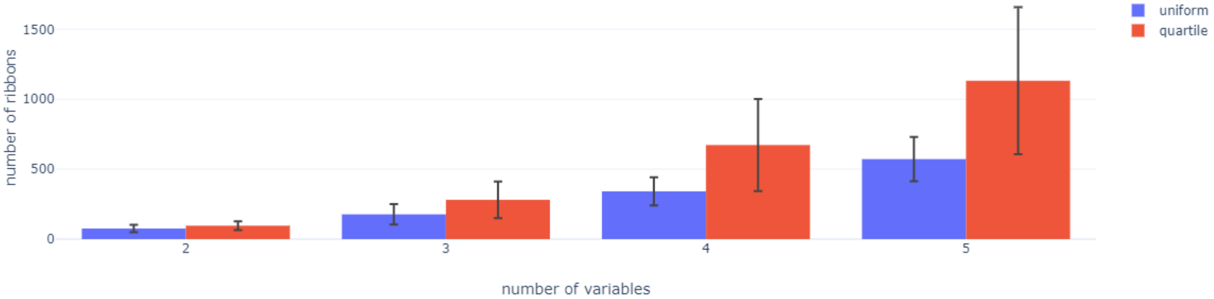


Figure 5.1: Bar plot showing number of variables vs. the number of ribbons (blue bar for uniform categorization and red bar for quartile categorization).

We realized that quartile categorizing creates more ribbons, which brings more ribbon crossing (Figure 5.2) and statistics columns rendering time (Figure 5.4). But it does not seem to affect the Parallel Set rendering time a lot. Also, even though different categorization methods have different complexity levels for the same measurement configuration, it does not seem to affect the trends of each measurement value for increasing

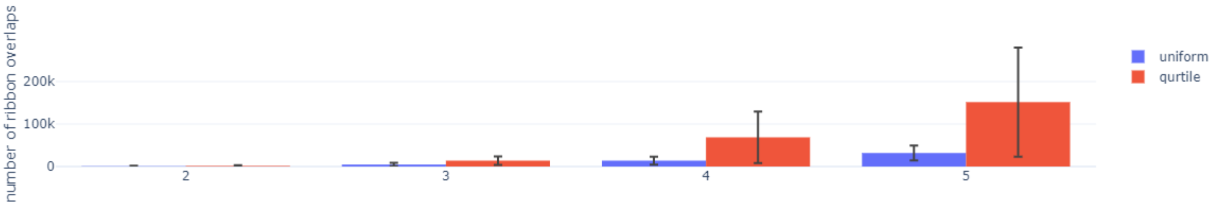


Figure 5.2: Bar plot showing the number of variables vs. the number of ribbon overlaps (blue bar for uniform categorization and red bar for quartile categorization).

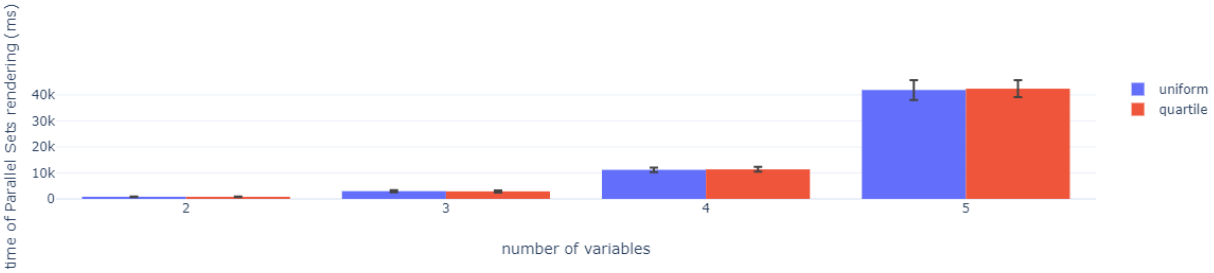


Figure 5.3: Bar plot showing the number of variables vs. the time of Parallel Sets rendering (blue bar for uniform categorization and red bar for quartile categorization).

variable counts. For the visual complexity, we found that both ribbon count (Figure 5.1) and ribbon crossing count (Figure 5.2) show exponential increments. For the time complexity, we found that only the rendering time of the Parallel Sets (Figure 5.3) shows exponential increment tendency; for the time of rendering statistics columns (Figure 5.4) and time of categorization (Figure 5.6), it shows a relatively linear trend; for the data querying time (Figure 5.5), it is relatively constant. We also noticed that, when the variable count becomes large (such as 5 in our case), the rendering time of the Parallel Sets is the only part having an exponential increment trend and it takes most of the time (over 98%).

When picking up less than 5 variables, it usually takes just a few seconds to load the visualization. However, when the variable count is equal to or greater than 5, the time of rending would dramatically increase to the level of minutes. In this process, we also found that the bottleneck is the rendering of the Parallel Sets as the ribbon count increases significantly with an increasing variable count. Thus, our suggestion is to use Set-Stat-Map with less than 5 variables at one time to balance the need of getting information and performance. In addition, applying the category number limitation and merging unimportant ones for axes is recommended since it significantly reduces the total ribbons to be displayed, which could potentially reduce the rendering time and the cognitive load.

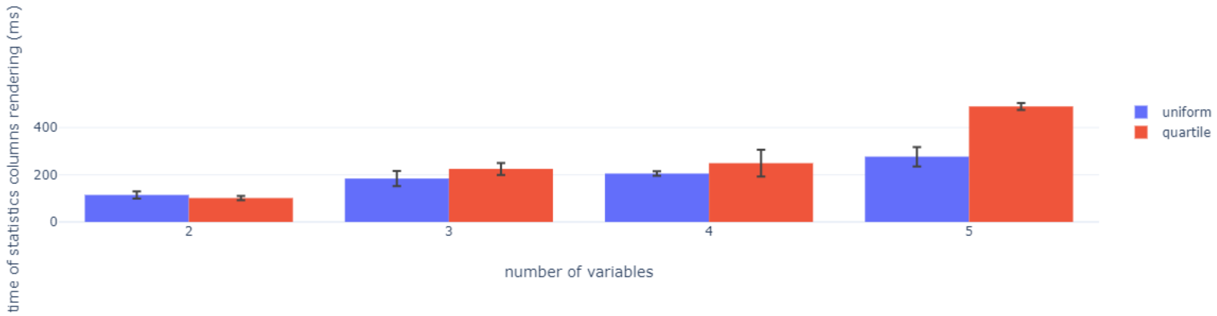


Figure 5.4: Bar plot showing the number of variables vs. the time of statistics columns rendering (blue bar for uniform categorization and red bar for quartile categorization).

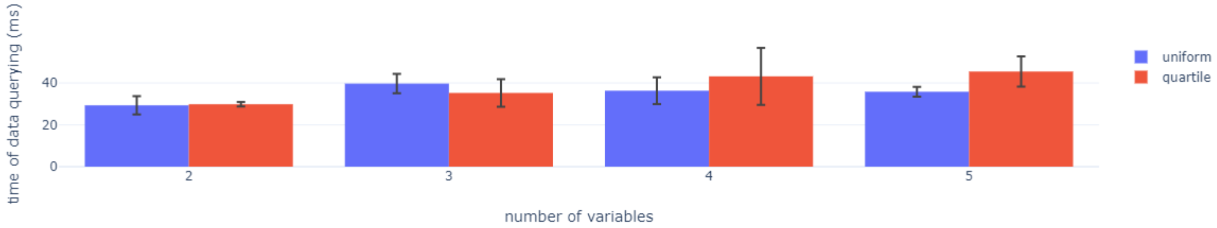


Figure 5.5: Bar plot showing the number of variables vs. the time of data querying (blue bar for uniform categorization and red bar for quartile categorization).

5.2 Use Cases

To evaluate the usability of the Set-Stat-Map, we take multiple use cases with datasets from totally different areas, which we described in the previous chapter. For the meteorological dataset, the experts used our system and gave us feedback over several development iterations; for the Airbnb dataset, a user participated to explore our system and complete some given tasks.

5.2.1 Meteorological Dataset

We illustrate the usage of the system by describing a use case that was run by the experts who are research scientists and have more than ten years of experience in this field (Figure 5.7). The experts used a hydroclimatic data from 2013 for the Canadian Prarieis Ecozone (CPE). The CPE is cropped by a polygon covering an area of approx. 700,000 sq km (270,000 sq miles). Users can select a subregion of interest using a mouse drag interaction.

Workflow: The experts selected the min temp, precipitation, and wind. Since the order of the variables in this study matters, they focused on the first two variables selected (i.e. min temp and precipitation). They chose the quartiles view as this is more common in hydrology to look at the highest and lowest quartiles.

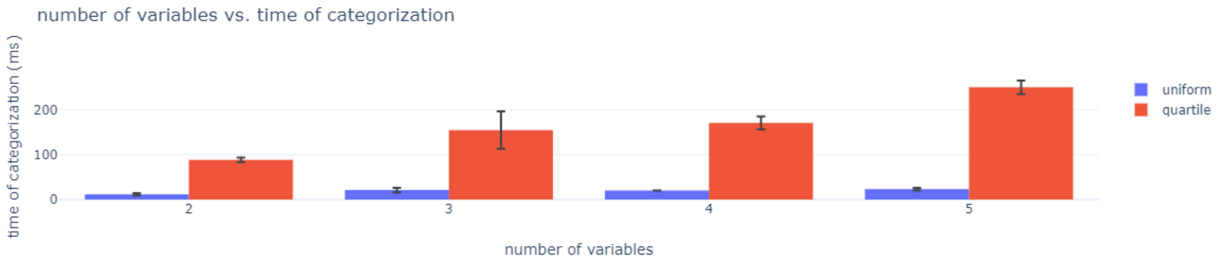


Figure 5.6: Bar plot showing the number of variables vs. the time of categorization (blue bar for uniform categorization and red bar for quartile categorization).

Table 5.1: Time distribution of different potential time-consuming parts for different variable counts. (the displayed times are rounded to the nearest ms)

height Num. of var.		Rendering Par. Sets	Rendering Stat. col.	Query	Categorization
2	time (ms)	816	108	30	50
	percentage	81.34%	10.75%	2.95%	4.97%
3	time (ms)	2929	204	38	88
	percentage	89.88%	6.27%	1.15%	2.69%
4	time (ms)	11277	227	40	95
	percentage	96.89%	1.95%	0.34%	0.82%
5	time (ms)	42127	383	41	136
	percentage	98.69%	0.90%	0.10%	0.32%

Looking at the Parallel Sets, they could immediately view that the lowest and highest quartiles of min temp are related to lower and higher amount of precipitation, respectively. They could also tell that higher min temp that corresponds with warmer months are associated with less wind. While colder months are expected to be with stronger wind gusts.

The texture showed more precipitation is associated with less windy scenarios and warmer months. The bar plot showed the monthly distribution of each selected variable. Precipitation is generally higher in the month of July but its daily values are with the highest variability. The average monthly wind gust and its daily variability in the months of January and November also are the highest. The background of the bar plots in each column was identical and showed the distribution of the first selected variable, in this case, min temp with blue being the lower quantile or dark blue being the most dominant color in January and December and dark brown being the highest minimum temperature in the summer months and September. Finally, the Google overlay map showed the spatial distribution of the min temperature in color and precipitation in texture for a given month. They selected the month January and noticed that for 2013, Southern Manitoba, east, and north Saskatchewan experienced lower temperatures than south Saskatchewan and Alberta.

The experts mentioned that the series of visualization provided by the system can be valuable for comparing key hydroclimatic variables. They also mentioned that the system, however, has its spatial and temporal limitations: (a) The data here is only collected for the year 2013, which makes it difficult to have meaningful comparisons with other years, comparing trends, and changes, which would be of great interest for hydrologists and for climate change research. (b) The spatial resolution of the data is coarse (36 km) which is a challenge in understanding changes and shifts in variables over farms or urbanized areas.

We asked the experts about what tools they generally use to visualize geo-analysis results and how Set-Stat-Map would compare to them. They mentioned that a typical output involves a single variable color-coded map, generated in Python or R, that can be also overlaid on a static or dynamic Google Maps. Generally, a couple of such maps, from different periods of time with the same variable or different variables from the same period of time, are compared side by side for decision making. In comparison, Set-Stat-Map is capable of showing more than one variable in one geographic location at once making the relations among variables more detectable. The experts were enthusiastic about the prospect of Set-Stat-Map and using it as a tool to display various standardized hydro-climate indices at once. These indices are built over long-term hydrological variables and traditionally, they are more indicative of drought and pluvial conditions [51]. Therefore, for future work, the experts wanted to extend our focus on using Set-Stat-Map to present and compare the long-term indices, such as Standardized Precipitation Index (SPI) and Soil Moisture Deficit Index (SMDI), on one map. Another opportunity is to integrate Set-Stat-Map with existing hydrological models so that one can validate the models' prediction outputs against observed data.

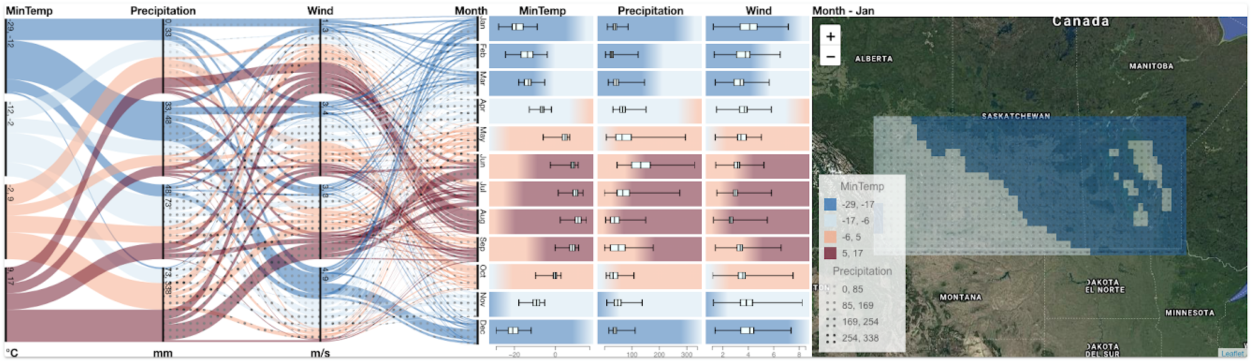


Figure 5.7: A Set-Stat-Map visualization for a meteorological dataset over Canadian Prairies.

5.2.2 Online Vacation Rental Dataset (Airbnb)

In this use case, we examine an Airbnb dataset, which is an online vacation rental dataset. The final processed data contained many categorical and numeric attributes, e.g., room type, longitude, latitude, bedrooms, administrative division, neighborhood, overall satisfaction, price, etc. Since this dataset contains both categorical and numerical data, it provides a good opportunity to demonstrate the functionality of Set-Stat-Map in handling mixed data types. To provide an idea of the usability of the system, here we report our interaction with a web developer with some knowledge of the Airbnb vacation rental process.

Workflow: In this case, the dataset was explored by an Airbnb user who has been working in web development industry for over two years and has a Bachelor degree in Computer Science. He had no experience with Parallel Sets visualization technique and took some time to understand and get used to it. Once he become familiar with the Parallel Sets and our basic design of the system, he started to explore the visualization in the aspect of a tenant who wants to find an accommodation for his next vacation and a potential renter who wants to list some of the extra rooms.

He picked room type (entire place, private room, or shared room), bedrooms (number of rooms), borough (the city division), and price (for a night) as attributes to be visualized in the Parallel Sets view. For the numeric attribute (price), he selected "quantile" as the categorization method in order to get even portions of segments, which he feels more comfortable to see. Then he chose reviews (the review count) and overall satisfaction (the rating out of five) as the attributes to be visualized in the statistics columns. After that, he made sure the room type attribute is on the first axis and the number of bedroom attribute is on the second axis because he wanted to see both room type (indicated with color) and the number of bedrooms (indicated with textures) on the map.

He first wanted to know if there are adequate number of options available for the lowest price range. He examined the color and texture combinations and quickly realized that, for the lowest price range, there is almost no entire place available and most of the available choices are all one-bedroom private rooms. He wanted to further investigate the spatial distribution of these available options, so he clicked the corresponding axis segment to show all these options' location distribution directly on the map view (Figure 5.8). The map view surprisingly shows that the options are spreading across the whole city including the city center (where he thought it would have just a few options in the price range). He concluded that if privacy is not a big concern (do not require the entire place), the lowest price range has a large space of selection.

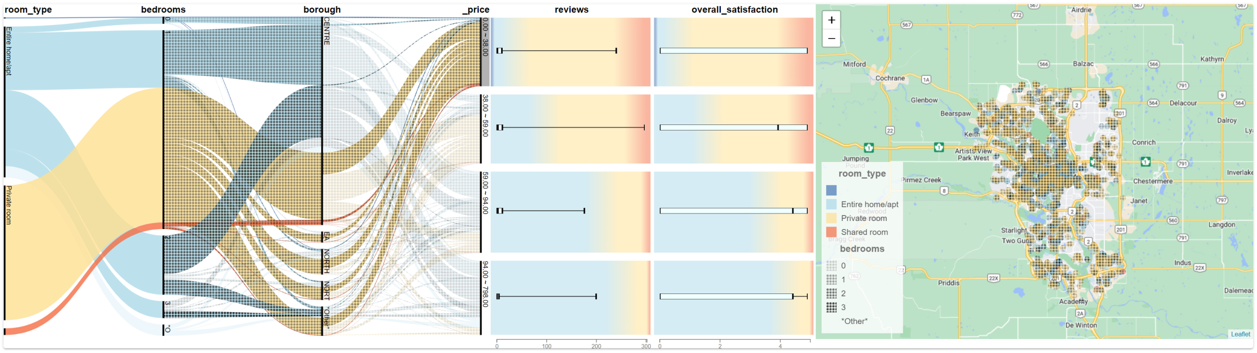


Figure 5.8: Exploration of Airbnb dataset with a selection of lowest price range.

He then tried to find what kind of accommodations are most welcomed by potential tenants. He made the last axis (price) ordering by the overall satisfaction in a descending manner (Figure 5.9). He also checked the reviews (review count) and realized that there are no obvious differences in distribution to make sure the review count is not a big factor in the actual overall satisfaction ratings. Then, he found that people feel more satisfied with accommodations that provide an entire place and are located near the city center or main roads in the price range from $59 to $94.

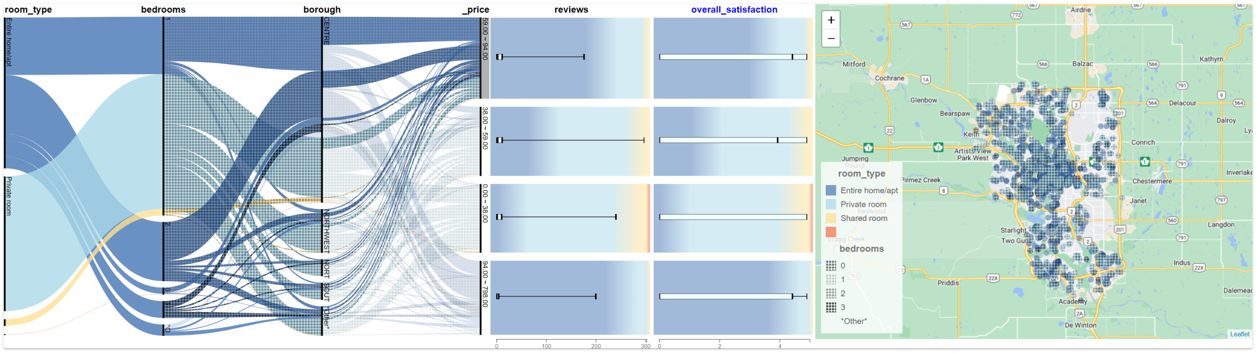


Figure 5.9: Exploration of Airbnb dataset with a selection of most satisfied price range.

5.2.3 Q&A Website for Programmers (StackOverflow)

StackOverflow [10] is a well-known question-and-answer website for programmers. In this use case, we look into a dataset of all users' posts in StackOverflow from 2008 to 2018. We worked with a software engineering researcher to create such a dataset of interest. Each data record contains a user ID, topic of the post, year of the post, and country of the developer.

The primary goal to use this dataset is to examine whether Set-Stat-Map can readily provide us with some interesting insights about the users' activities or posts. After creating the Set-Stat-Map visualization for this dataset we realized that the visualization reveals natural properties of the datasets.

Figure 5.10 shows an example of visualizing the StackOverflow dataset using Set-Stat-Map. In this example, we picked "Location", "Topic", and "Year" as the Parallel Sets axes; "United States", "United Kingdom", "Canada", and "Australia" are picked as our focus for locations; "sql", "java", and "html" are picked as our focus for topics. The visualization clearly shows the ratio of developers from different countries and the popularity of topics. However, our raw data only tracks the same group of users, who are leaving over time, which makes the total post count to decrease year by year. This trend can be easily seen from the last axis "year" of the Parallel Sets view, whose segments get less and less heights with increasing years. In Figure 5.10, the statistics columns view shows information about active year count, which counts how many years a user is active on StackOverflow (if a user has at least one post in a year, then the dataset considers it is an active year). We can see there is a trend of people having higher active year count over years, which is generally considered expected behaviour. In other words, because people are leaving, the remaining users must have higher and higher active year counts. The heatmap view should give us information about the popularity of a topic over the year, yet we see all of our selected topics are getting less and less popular, which might be due to the users getting more and more familiar with a topic and having less need to ask a question. Instead of getting a topic's post count over the year, the heatmap view might provide some ideas about which technology is more difficult (e.g., people keep asking questions). In this example, Java and SQL have almost the same level of post count at the beginning, but Java's post count drops much slower than SQL, which might indicate differences among these programming languages in different properties such as interest level, learning rate, user community, etc.

Set-Stat-Map visualization for the StackOverflow dataset shows some useful information. For example, from the statistics columns view in Figure 5.10, we can see that the range of active year of HTML and Java are quite different comparing to SQL at the end of the year. We can also see that, in the heatmap view, even though all three topics got less and less attentions, Java has the slowest decrease speed, which is likely because Java is relatively harder to master than the other two technologies.

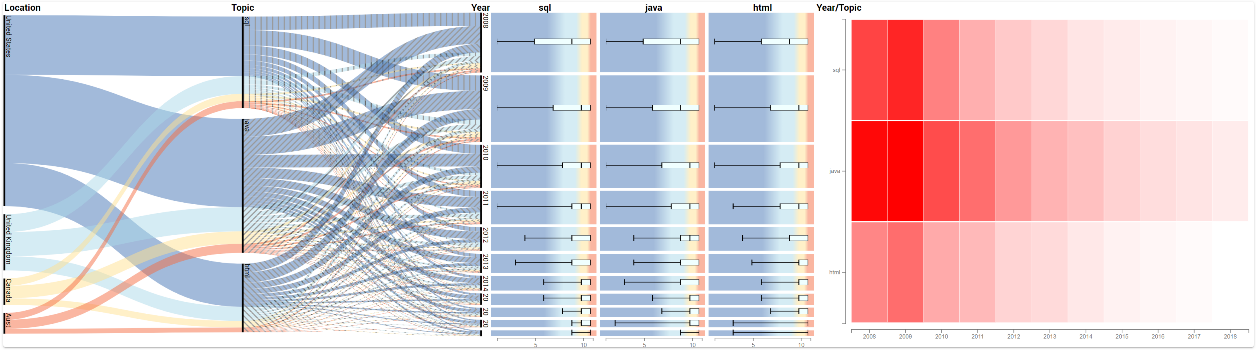


Figure 5.10: Visualizing a StackOverflow dataset using Set-Stat-Map.

5.3 Summary

In this chapter, we described the scalability, generalizability, and usability of Set-Stat-Map. For the performance, we run multiple test cases to gather the statistics of performance and then plot them for analysis. The results show that the ribbon count, ribbon overlaps, and rendering time all increase dramatically with the increased variable count. Based on the results, we give our suggestion of only choosing no more than 5 variables at the same time for the visualization, which ensures both the render time and the ribbon visibility. We also realize that the Parallel Sets view rendering takes the most proportion of the time (more than 98% when choosing five variables). Thus, in the future, optimizing the techniques of rendering the Parallel Sets view might largely improve Set-Stat-Map's scalability. For usability, we use Set-Stat-Map to visualize three different types of real-world datasets (Meteorological, Airbnb, StackOverflow) and it shows interesting insights for the Meteorological and Airbnb datasets but not for the StackOverflow dataset. It reminds us that to better use Set-Stat-Map, we need to carefully choose the dataset and design the visualization details.

6 Discussion

From our experience, we have seen Set-Stat-Map perform very well if both the number of axes (variables) and the number of intervals (categories) per variable are bounded by 5. For an axis to be useful, it needs to have at least two intervals, each corresponding to a reasonable number of data records. Since the ribbons are likely to get split in every subsequent axes, bringing the axes with a few categories earlier in the axes sequence may reduce the overplotting problem. It would be interesting to further investigate such conditions for a dataset to be effectively visualized using Set-Stat-Map. In addition, we have identified a number of key factors that need to be carefully considered when designing a Set-Stat-Map visualization.

Choice of axis: To effectively use Set-Stat-Map, users need to understand the concept that the color and texture come from the first and second axis, respectively, and the categories to be compared should be chosen as the last axis of the Parallel Sets.

One option to ease the learning process, especially for non-scientific users, is to ask to identify important categories. For example, after loading the data, the system can ask to choose a categorical attribute (independent variable) over which the users would like to see the data pattern. This will be the last axis of the Parallel Sets view. After the user chooses the attribute, the system can ask to select the categorical and numerical attributes (dependent variables) of interest to populate the Parallel Sets and statistics column views.

There can still be a steep learning curve to effectively use the numerical attribute as a categorical one. In the future, it would be interesting to investigate users' learning rate affects knowledge discovery.

Colours and opacity: One must add some transparency to the ribbons to provide users with a sense of connectedness. This also allows users to better perceive the line patterns that appear between adjacent axes. In a poorly designed color map, some ribbon colours may interfere with its texture. A systematic way to choose colours would be to choose a dark color for texture and light colours for the ribbon colour map, or vice versa. In our work, we colour the texture in gray and choose at most 8 other light colours from a perceptual colour map [2]. Ordering of the colours is also important as it helps users better understand the distribution of numeric ranges.

Choice of texture: Ribbon texture is an important aspect of Set-Stat-Map design. While examining the ribbons intersecting a line interval, the textures of the ribbons can reveal the composition of the second axis categories. In a poorly designed texture map, thin ribbons can sometimes entirely miss the pattern or provide a partial pattern creating ambiguity. An example of the former problem is when a dot pattern with varying density is used as texture. The problem of cut-off is illustrated in Figure 6.1(left), where the top ribbon can be interpreted as a cut-off cross or wave pattern instead of the x shape. Similarly, The fourth ribbon from the top can be seen as a cut-off circle or wave pattern instead of the circle shape.

Another problem with choosing complex patterns is that they are difficult to relate to numeric ranges. Furthermore, overlapping ribbons makes the visualization cluttered by creating noisy patterns, as illustrated in Figure 6.1(right), which increases users' cognitive load.

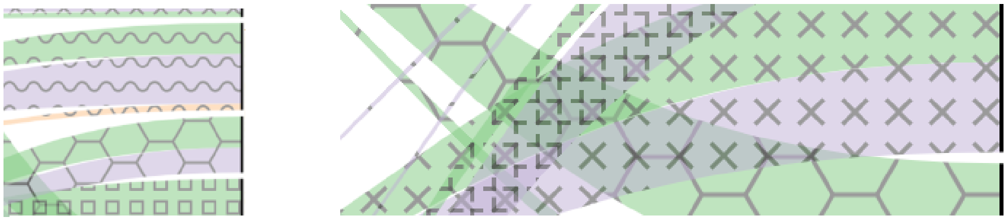


Figure 6.1: (left) Texture cut-off issues. (right) noisy pattern created by a combination of overlapping textures.

Adaptive label size and mouse over: A Set-Stat-Map design creates intervals of widely varied lengths. Therefore, some intervals may be very thin, creating a problem for label placement, e.g., see Figure 6.2(left). To best leverage the vertical space, a careful design must attempt to limit the number of categories per axis that are visible simultaneously, and provide functionality such that the users can sort and scroll the category list. Furthermore, the label size should be adaptive to the length of the line to avoid visual occlusion (Figure 6.2(middle)), and allow users to inspect detailed information on mouse over (Figure 6.2(right)).

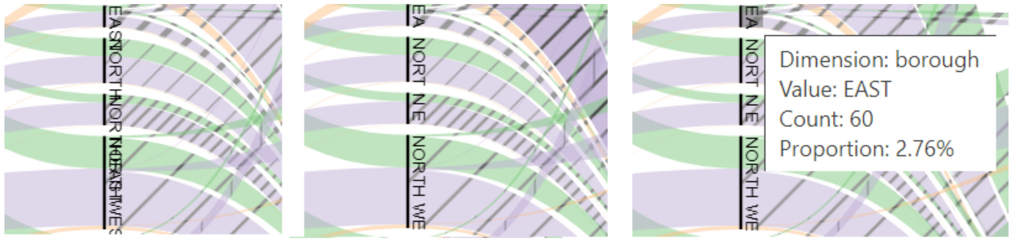


Figure 6.2: (left) Label overlapping. (middle) Adaptive labelling. (right) Mouseover interaction.

Linear gradient color: The background of each cell in the statistics columns view is colored with a linear gradient. Such a linear gradient may not create a color range with a proper perceptual balance, i.e., people may perceive the width or strength of the colors differently than the underlying numeric values.

One problem with multiple colors appears to be with numeric ranges, where the preferred color map is sequential or diverging. Since the adjacent colors in such a color map are perceptually close to each other, the linear gradient may make it harder to distinguish the color boundaries (Figure 6.3). For categorical color maps, one may obtain a background with multiple vertical stripes. We suggest minimal use of the number of colors since multiple vertical stripes may give rise to unwanted visual effects (e.g., McCollough effect [27] or optical illusion) over prolonged use. Further investigation on how humans perceive such linear gradients in a visual analytics context, or whether the benefit of gradient exceeds the potential risk can be an exciting avenue of research.

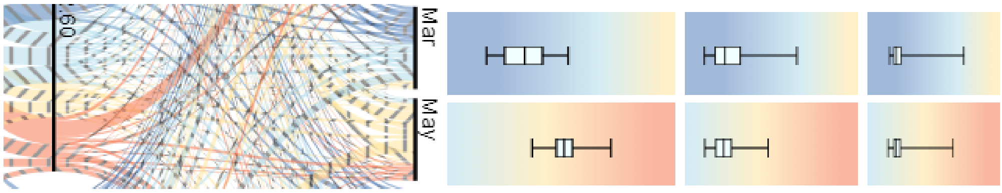


Figure 6.3: Linear gradient makes the boundaries harder to distinguish.

Adaptive heatmap: The Set-Stat-Map visualization uses a heatmap view on the map, where the cells are colored and textured based on the first and second axis. If both these axes are numerical, then for each cell one can choose the color and text based on the average value. However, for categorical attributes, it needs to be based on a summary statistic. In our design, we pick the color and texture based on the highly frequent categories. However, based on the data (e.g., when all categories are equally frequent) this may create misleading representation. One may partially improve the problem by using an adaptive quadtree-based heatmap [26] or by providing complex glyphs (e.g., pie charts). However, a conscious investigation is required to shed more light on this situation.

Perceptually-motivated Parallel Sets: Our effort for stylizing ribbons to analyze the distribution of a combination of attributes can be extended further by leveraging the idea of perceptually-motivated visualization [36] by mapping the categories of different axes with different perceptual properties. Effective perceptually-motivated visualization is harder to design and also to interpret. However, such visualization can still be valuable in an infographic setting, as well as to examine or extend the boundary of human perception.

6.1 Summary

Although we argue that Set-Stat-Map is useful to visualize multi-dimensional datasets with mixed data types (numerical, categorical, and spatial), a poor design could ruin it. We need to be very careful to choose the axes, colours, textures, label size, etc. Also, we should avoid using linear colour gradients for the statistics column backgrounds because that could make the boundaries harder to distinguish. The styles for the map view are also an important aspect to be carefully considered as they can cause higher cognitive loads and misunderstandings. In the future, Set-Stat-Map still has some opportunities to be improved. For example, we can augment the categories of different axes with different perceptual properties for the Parallel Sets view to leverage the idea of perceptual visualization.

7 Conclusion

Although Parallel Coordinates and Parallel Sets visualizations are widely used to interactively explore numerical and categorical datasets, respectively, visualization of spatial data with mixed numeric and categorical attributes is not yet well explored. To solve such a problem, we propose Set-Stat-Map that leverages the idea of multiple coordinated views. The design harmoniously blends an enhanced Parallel Sets view, a statistics columns view, and a map view to visualize mixed data in a linear fashion. Our design allows users to easily explore the pairwise attribute relationships, statistical summaries and trends, and spatial distribution of various attributes.

We demonstrated the functionality of Set-Stat-Map using real-life datasets. The meteorological dataset shows how our visualization can help an environment scientist find interesting trends in meteorology; the Airbnb dataset shows how our system can help a user to find the best match of housing listings on Airbnb; the StackOverflow dataset exposes a bad use case for our purpose.

We also evaluated the performance of Set-Stat-Map. The bottleneck of the system is rendering the Parallel Sets view when the ribbon count becomes too large due to a larger variable count. So we provided a suggestion of picking no more than 5 variables at a time to get the best balance of obtaining information and performance. We also showed that set merging for axes could be a good idea for scalability and ease of understanding the visualization.

Even though we found the potential of Set-Stat-Map for visualizing complex multi-dimensional datasets, we showed examples when it may not reveal interesting insights without a careful design. We pointed out some key factors to be considered when designing a Set-Stat-Map visualization.

Although Set-Stat-Map is currently a research prototype, future work may provide a web-based implementation available to developers through open-source software repositories. We believe our approach to visualizing mixed data types will inspire future work in visualization and interface design to effortlessly explore heterogeneous datasets.

References

[1] airbnb-data – tom slee. https://tomslee.net/category/airbnb-data.

[2] Colorbrewer: Color advice for maps. www.ColorBrewer.org.

[3] D3.js - data-driven documents. https://d3js.org/.

[4] Global weather data for swat. https://globalweather.tamu.edu/.

[5] Manifold learning on handwritten digits: Locally linear embedding, isomap.... https://scikit-learn.org/stable/auto_examples/manifold/plot_lle_digits.html#sphx-glr-auto-examples-manifold-plot-lle-digits-py.

[6] Parallel coordinates. https://github.com/unsetbit/parallel-coordinates-chart.

[7] Parallel sets. https://www.jasondavies.com/parallel-sets/.

[8] Sqlite. https://www.sqlite.org.

[9] sql.js. https://sql.js.org/.

[10] Stackoverflow. https://stackoverflow.com/.

[11] Stenciljs. https://stenciljs.com/.

[12] tomslee/airbnb-data-collection: Data collection for airbnb listings. https://github.com/tomslee/airbnb-data-collection.

[13] Venn diagram. https://en.wikipedia.org/wiki/Venn_diagram.

[14] Geo-coordinated parallel coordinates (gcpc): Field trial studies of environmental data analysis. *Visual Informatics*, 2(2):111 – 124, 2018.

[15] Hervé Abdi and Lynne J Williams. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459, 2010.

[16] Basak Alper, Nathalie Henry Riche, Gonzalo A. Ramos, and Mary Czerwinski. Design study of linesets, a novel set visualization technique. *IEEE Trans. Vis. Comput. Graph.*, 17(12):2259–2267, 2011.

[17] Bilal Alsallakh, Luana Micallef, Wolfgang Aigner, Helwig Hauser, Silvia Miksch, and Peter J. Rodgers. Visualizing sets and set-typed data: State-of-the-art and future challenges. In Rita Borgo, Ross Maciejewski, and Ivan Viola, editors, *16th Eurographics Conference on Visualization, EuroVis 2014 - State of the Art Reports*. Eurographics Association, 2014.

[18] Bilal Alsallakh, Luana Micallef, Wolfgang Aigner, Helwig Hauser, Silvia Miksch, and Peter J. Rodgers. The state-of-the-art of set visualization. *Comput. Graph. Forum*, 35(1):234–260, 2016.

[19] Michelle Q. Wang Baldonado, Allison Woodruff, and Allan Kuchinsky. Guidelines for using multiple views in information visualization. In Vito Di Gesù, Stefano Levialdi, and Laura Tarantino, editors, *Proceedings of the working conference on Advanced visual interfaces (AVI)*, pages 110–119. ACM Press, 2000.

[20] J. Blaas, C. Botha, and F. Post. Extensions of parallel coordinates for interactive exploration of large multi-timepoint data sets. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1436–1451, 2008.

[21] Lawrence A Bruckner. On chernoff faces. In *Graphical representation of multivariate data*, pages 93–121. Elsevier, 1978.

[22] Stirling Christopher Chow. Generating and drawing area-proportional euler and venn diagrams. 2007.

[23] Christopher Collins, Gerald Penn, and Sheelagh Carpendale. Bubble sets: Revealing set relations with isocontours over existing visualizations. *IEEE Trans. Vis. Comput. Graph.*, 15(6):1009–1016, 2009.

[24] Michael A. A. Cox and Trevor F. Cox. *Multidimensional Scaling*, pages 315–347. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[25] Shamse Tasnim Cynthia, Banani Roy, and Debajyoti Mondal. Feature transformation for improved software bug detection models. In *15th Innovations in Software Engineering Conference*, pages 1–10, 2022.

[26] Cicero Augusto de Lara Pahins, Sean A. Stephens, Carlos Scheidegger, and João Luiz Dihl Comba. Hashedcubes: Simple, low memory, real-time visual exploration of big data. *IEEE Trans. Vis. Comput. Graph.*, 23(1):671–680, 2017.

[27] Peter C. Dodwell and G. Keith Humphrey. A functional theory of the mccollough effect. *Psychological Review*, 97(1):78–89, 1990.

[28] Danny Dorling, Anna Barford, and Mark Newman. Worldmapper: the world as you've never seen it before. *IEEE transactions on visualization and computer graphics*, 12(5):757–764, 2006.

[29] David A Ellsworth, Christopher E Henze, and Bron C Nelson. Interactive visualization of high-dimensional petascale ocean data, 2017.

[30] Jared Espenant and Debajyoti Mondal. Streamtable: An area proportional visualization for tables with flowing streams. In Petra Mutzel, Md. Saidur Rahman, and Slamin, editors, *WALCOM: Algorithms and Computation*, pages 97–108. Springer, 2022.

[31] William Evans, Stefan Felsner, Michael Kaufmann, Stephen G. Kobourov, Debajyoti Mondal, Rahnuma Islam Nishat, and Kevin Verbeek. Table cartogram. *Computational Geometry*, 68:174–185, 2018.

[32] Jean Flower, Andrew Fish, and John Howse. Euler diagram generation. *J. Vis. Lang. Comput.*, 19(6):675–694, 2008.

[33] Samuel Gratzl, Alexander Lex, Nils Gehlenborg, Hanspeter Pfister, and Marc Streit. Lineup: Visual analysis of multi-attribute rankings. *IEEE Trans. Vis. Comput. Graph.*, 19(12):2277–2286, 2013.

[34] Mohammad Rakib Hasan, Debajyoti Mondal, Jarin Tasnim, and Kevin A. Schneider. Putting table cartograms into practice. In *Advances in Visual Computing: 16th International Symposium, ISVC*, page 91–102. Springer-Verlag, 2021.

[35] Christopher G Healey and James T Enns. Large datasets at a glance: Combining textures and colors in scientific visualization. *IEEE transactions on visualization and computer graphics*, 5(2):145–167, 1999.

[36] Christopher G. Healey and James T. Enns. Attention and visual memory in visualization and computer graphics. *IEEE Trans. Vis. Comput. Graph.*, 18(7):1170–1188, 2012.

[37] Julian Heinrich and Daniel Weiskopf. State of the art of parallel coordinates. In Mateu Sbert and László Szirmay-Kalos, editors, *Proc. of the 34th Annual Conference of the European Association for Computer Graphics (Eurographics)*, pages 95–116. Eurographics Association, 2013.

[38] Michael C. Hout, Megan H. Papesh, and Stephen D. Goldinger. Multidimensional scaling: Multidimensional scaling. *Wiley Interdisciplinary Reviews: Cognitive Science*, 4(1):93–103, 2013.

[39] Alfred Inselberg and Bernard Dimsdale. Parallel coordinates: A tool for visualizing multi-dimensional geometry. In Arie E. Kaufman, editor, *1st IEEE Visualization Conference, IEEE Vis*, pages 361–378. IEEE Computer Society Press, 1990.

[40] Robert Kosara, Fabian Bendix, and Helwig Hauser. Parallel sets: Interactive exploration and visual analysis of categorical data. *IEEE Trans. Vis. Comput. Graph.*, 12(4):558–568, 2006.

[41] Alexander Lex, Nils Gehlenborg, Hendrik Strobelt, Romain Vuillemot, and Hanspeter Pfister. Upset: Visualization of intersecting sets. *IEEE Trans. Vis. Comput. Graph.*, 20(12):1983–1992, 2014.

[42] Fang Liu and Rosalind W Picard. Periodicity, directionality, and randomness: Wold features for image modeling and retrieval. *IEEE transactions on pattern analysis and machine intelligence*, 18(7):722–733, 1996.

[43] Debajyoti Mondal, Manishankar Mondal, Chanchal K. Roy, Kevin A. Schneider, Yukun Li, and Shisong Wang. Clone-world: A visual analytic system for large scale software clones. *Vis. Informatics*, 3(1):18–26, 2019.

[44] Manishankar Mondal, Chanchal K Roy, and James R Cordy. Nicad: A modern clone detector. In *Code Clone Analysis*, pages 45–50. Springer, 2021.

[45] Saikat Mondal. Created from stackoverflow data dump. Personal Communication.

[46] Tamara Munzner. *Visualization Analysis and Design.* A.K. Peters visualization series. A K Peters, 2014.

[47] Kristin Potter, Hans Hagen, Andreas Kerren, and Peter Dannenmann. Methods for presenting statistical information: The box plot. In *VLUDS*, pages 97–106, 2006.

[48] J. C. Roberts. State of the art: Coordinated multiple views in exploratory visualization. In *Fifth International Conference on Coordinated and Multiple Views in Exploratory Visualization (CMV 2007)*, pages 61–71, 2007.

[49] Geraldine E. Rosario, Elke A. Rundensteiner, David C. Brown, Matthew O. Ward, and Shiping Huang. Mapping nominal values to numbers for effective visualization. *Inf. Vis.*, 3(2):80–95, 2004.

[50] M. Joan Saary. Radar plots: a useful way for presenting multivariate health care data. *Journal of Clinical Epidemiology*, 61(4):311–317, 2008.

[51] Sara Sadri, Ming Pan, Yoshihide Wada, Noemi Vergopolan, Justin Sheffield, James S. Famiglietti, Yann Kerr, and Eric Wood. A global near-real-time soil moisture index monitor for food security using integrated SMOS and SMAP. *Remote Sensing of Environment*, 246(7):111864, 2020.

[52] Manojit Sarkar, Scott S. Snibbe, Oren J. Tversky, and Steven P. Reiss. Stretching the rubber sheet: A metaphor for viewing large layouts on small screens. In Scott E. Hudson, Randy Pausch, Brad T. Vander Zanden, and James D. Foley, editors, *Proceedings of the Sixth ACM Symposium on User Interface Software and Technology, UIST*, pages 81–91. ACM, 1993.

[53] Gaurav Sharma and Raja Bala. *Digital color imaging handbook.* CRC press, 2017.

[54] Haleh H Shenas and Victoria Interrante. Compositing color with texture for multi-variate visualization, 2005.

[55] Ben Shneiderman. Tree visualization with tree-maps: 2-D space-filling approach. *ACM Trans. Graph.*, 11(1):92–99, 1992.

[56] Hideyuki Tamura, Shunji Mori, and Takashi Yamawaki. Textural features corresponding to visual perception. *IEEE Transactions on Systems, man, and cybernetics*, 8(6):460–473, 1978.

[57] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[58] Laurens Van Der Maaten, Eric Postma, Jaap Van den Herik, et al. Dimensionality reduction: a comparative. *J Mach Learn Res*, 10(66-71):13, 2009.

[59] Shisong Wang, Debajyoti Mondal, Sara Sadri, Chanchal K Roy, James S Famiglietti, and Kevin A Schneider. Set-stat-map: Extending parallel sets for visualizing mixed data. In *2022 IEEE 15th Pacific Visualization Symposium (PacificVis)*, pages 151–160. IEEE, 2022.

[60] Colin Ware. *Information Visualization, Second Edition: Perception for Design (Interactive Technologies)*. 2004.

[61] David F Williamson, Robert A Parker, and Juliette S Kendrick. The box plot: a simple visual method to interpret data. *Annals of internal medicine*, 110(11):916–921, 1989.

[62] Kent Wittenburg, Tom Lanning, Michael Heinrichs, and Michael Stanton. Parallel bargrams for consumer-based information exploration and choice. In Joe Marks and Elizabeth D. Mynatt, editors, *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology, UIST*, pages 51–60. ACM, 2001.

[63] Keqin Wu and Song Zhang. Visualizing 2D scalar fields with hierarchical topology, 2015.